2.35 PROBABILITY OF HIT

Probability of hit (P_h) is one of the primary measures of effectiveness for anti-aircraft artillery (AAA) systems. The probability of a projectile hitting a target is a function of their relative positions at the closest point of approach (CPA), the presented area of the target in a plane normal to the flight path of the round, and the dispersion associated with the gun and its ammunition.

In the sensor system autotrack mode, the fire control computer (FCC) provides an estimate of the necessary lead and super elevation angles required to intercept a moving target after an estimated projectile time of flight. In manual mode, the operator slews the guns to an aim point and fires either continuously (barrage fire) or in bursts in the specified direction. Once the target is within tactical range, bullets are fired at the specified firing rate of the system and travel along ballistic paths.

Errors inherent in gun systems cause the impact point to differ from the target centroid by an amount which is random from shot-to-shot. The total system error can be expected to contain a lead error, an aim error, and a round-to-round dispersion about the aim point. Lead errors, also called systemic errors, arise from sources such as imbalances in the servos, wrong gain settings in the amplifiers of the radar and fire control computer, inaccuracies in wind or air density estimates, etc. Their net effect is to impart a bias on the center of impact points; this bias is constant (but unknown) for a particular engagement, and varies from one engagement to the next. The aim errors vary with time during an engagement. They are due to changes in tube sag, incorrect radar tracking, changes in weather, etc. Round-to-round dispersion errors are caused by the combined effect of variation in shell manufacture, powder weight, moisture content, etc. Generally, insufficient quantitative data is available to describe any of these error variables, so they are usually assumed to be described by independent Gaussian (normal) distributions. Several methods have been used to combine these factors and the target presented area into a final P_h; in RADGUNS, probabilities are calculated via both the "Quasi-Combat Equation" and the "Salvo Equation" methodologies.

The intent of the Probability of Hit FE in RADGUNS is to calculate single shot, burst, and cumulative probabilities of the AAA gun round(s) hitting a target during an engagement.

2.35.1 Functional Element Design Requirements

This section presents the design requirements for the Probability of Hit FE in RADGUNS for the applicable radar system.

a. The Probability of Hit FE will calculate the probability of an AAA projectile (round) hitting a target as a function of their separation at the CPA, the presented area of the target and the dispersion of the ammunition.

b. The target will be approximated as an ellipsoid based on target length, wing span, and height.

c. The target orientation with respect to the AAA system at the CPA will be used to determine the contributions from each presented area (6- or 26-view

presented area data) to the composite elliptical area presented to the AAA system.

d. The major and minor axes of the target elliptical area presented to the AAA system (from requirement 3) will be used as the variances of the target area distribution calculation; the area will be represented by an elliptic Gaussian distribution.

e. The "Quasi-Combat Equation" will be used to calculate the P_h for a round.

f. The "Salvo Formula" will be available to calculate the P_h for a burst.

g. Cumulative P_h calculation will be available for multiple rounds for an entire engagement.

2.35.2 Functional Element Design Approach

This section contains descriptions of logic and algorithms used to implement the requirements of Section 2.35.1. These descriptions are grouped by design elements, which are specific components of the FE design.

Design Element 35-1: Time at Closest Point of Approach

The engagement geometry at the minimum distance between a round and a target is the subject of P_h calculations. When using finite time steps to move a target or projectile, the exact time at which the closest separation is achieved may not be identified with sufficient accuracy (for the purposes of P_h calculations). Therefore, if the target-round separation at each time step in a simulation is observed, the separation will decrease followed by an increase; the geometry at exactly the closest point of approach generally must be extrapolated from the geometry at the simulation time before the increase in separation is observed; the simulation time at the approximate time of closest approach is designated as t_{ap}. The method used to find the exact time period between t_{ap} and CPA, t_0, after the approximate time of closest approach is described next.

Linear motion of the shell is assumed between simulation time steps, so the position of the round and target at time t_0 relative to t_{ap} can be calculated in the weapon system frame as follows:

$$\bar{R}(t_0) = \bar{R}(t_{ap}) + \bar{V}_R(t_{ap})t_0 \qquad [2.35\text{-}1]$$
$$\bar{T}(t_0) = \bar{T}(t_{ap}) + \bar{V}_T(t_{ap})t_0$$

where:
- $\bar{R}$ = round position (m)
- $\bar{V}_R$ = round velocity (m/s)
- $\bar{T}$ = target position (m)
- $\bar{V}_T$ = target velocity (m/s)

The relative separation between the target and round at time t_0 can be calculated as:

$$\bar{R}(t_0) - \bar{T}(t_0) = (\bar{R}(t_{ap}) - \bar{T}(t_{ap})) + (\bar{V}_R(t_{ap}) - \bar{V}_T(t_{ap}))t_0 \qquad [2.35\text{-}2]$$

Substituting

$$\begin{aligned} \bar{D}(t_0) &= \bar{R}(t_0) - \bar{T}(t_0) \\ C &= (\bar{R}(t_{ap}) - \bar{T}(t_{ap})) \\ V &= (\bar{V}_R(t_{ap}) - \bar{V}_T(t_{ap})) \end{aligned} \quad [2.35\text{-}3]$$

Equation [2.35-2] can be written as:

$$\bar{D}(t_0) = \bar{C} + \bar{V}t_0 \quad [2.35\text{-}4]$$

Vector algebra can be used to find an expression for the round-target separation that is useful for finding t_0. The distance between the round and the target can be written as:

$$\begin{aligned} \bar{D}(t) &= (\bar{D}(t) \quad \overline{D(t)})^{\frac{1}{2}} \\ &= ((\bar{C} + \bar{V}t) \quad (\bar{C} + \bar{V}t))^{\frac{1}{2}} \\ &= (\bar{C} \quad \bar{C} + 2\bar{C} \quad \bar{V}t + \bar{V} \quad \bar{V}t^2)^{\frac{1}{2}} \end{aligned} \quad [2.35\text{-}5]$$

$\overline{D}(t_o)$ can be minimized by setting its first derivative to zero:

$$\frac{d\bar{D}(t_0)}{dt_0} = \frac{\bar{C} \quad V + \bar{V} \quad \bar{V}t_0}{\left(\bar{C} \quad \bar{C} + 2\bar{C} \quad \bar{V}t_0 + \bar{V} \quad \bar{V}t_0^2\right)^{\frac{1}{2}}} = 0 \quad [2.35\text{-}6]$$

The minimum distance between the round and the target occurs when the numerator of the above equation is zero:

$$\bar{C} \quad \bar{V} + \bar{V} \quad \bar{V}t_0 = 0 \quad [2.35\text{-}7]$$

Solving Equation [2.35-7], the time period required to attain CPA after the approximate time of closest approach is:

$$t_0 = \frac{-\bar{C} \quad \bar{V}}{\bar{V} \quad \bar{V}} \quad [2.35\text{-}8]$$

where: $\overline{C}$ = Position vector between target and round at t_{ap} (m)
$\overline{V}$ = Velocity vector between round and target at t_{ap} (m/s)

Design Element 35-2: Miss Distance

Miss distance (of the round hitting the target) at the CPA is calculated using the miss distance at time t_{ap}, the time period to CPA (t_0), and the relative velocities of the round and target. The components of the miss distance between the round and target at CPA are extrapolated from their current velocities at the approximate CPA and t_0 by using Equation [2.35-1]. The miss distance, D_m, is calculated using the Pythagorean Theorem:

$$D_m = (X^2 + Y^2 + Z^2)^{\frac{1}{2}} \quad [2.35\text{-}9]$$

where: X, Y, Z = Miss distance Cartesian components (m)

Design Element 35-3: Target Presented Area

Target aircraft presented area data often are available for either six or twenty-six target orientations. Most aircraft presented areas can be approximated by ellipses. For this design element, the shape of an ellipse is characterized by the ellipsoidal ratio which is the ratio of the major axis to the minor axis. Target height, width, and length can be used to determine the ellipsoidal ratio of the target area.

For the front and rear views of the target, the wing span is typically greater than the height of the body and is considered the major axis, *a*, of the ellipse. However, the wings are thin compared to the body thickness; thus, the body height is assumed to be the major contributor to the target area. The area, A, of an ellipse in terms of its minor axis, *b*, and ellipsoidal ratio, $\varepsilon = a/b$, is:

$$\begin{aligned} A &= \pi ab \\ &= \pi \varepsilon b^2 \end{aligned} \qquad [2.35\text{-}10]$$

Substituting the target presented area, A_T, for A and solving for the minor axis in terms of the ellipsoidal ratio yields:

$$b = \sqrt{\frac{A_T}{\varepsilon \pi}} \qquad [2.35\text{-}11]$$

For the side, top, and bottom views of the target, length is assumed to be the major contributor to the target presented area as well as the major axis. The area of an ellipse in terms of its major axis is:

$$\begin{aligned} A &= \pi ab \\ &= \frac{\pi a^2}{\varepsilon} \end{aligned} \qquad [2.35\text{-}12]$$

Substituting A_T for A and solving for the major axis yields:

$$a = \sqrt{\frac{A_T \varepsilon}{\pi}} \qquad [2.35\text{-}13]$$

A target of arbitrary orientation generally has apparent lengths of the major and minor axes that are different than the physical dimensions of the target; also, the target presented area generally is a combination of area data for specific aspect angles. In the horizontal (azimuthal) plane, unless the target is viewed nose-on or tail-on, or viewed directly out one of the wings, some part of either the front or rear of the target will be visible as well as some part of either the left or right side (see Figure 2.35-1). The target azimuthal aspect angle, φ, presented to the weapon system has a convention that $\varphi=0$ represents a front view, and $\varphi = \pi$ represents the rear view; the radian measures designated in Figure 2.35-1 denote the values of φ. In the vertical (elevation) plane, unless the target is viewed from directly out one of the wings, some portion of either the top or the bottom of the aircraft will be visible

(see Figure 2.35-2). The target elevation aspect angle, θ, presented to the weapon system has a convention that $\theta = 0$ represents a bottom view, and $\theta = \pi$ represents the top view; the radian measures designated in Figure 2.35-1 denote the values of θ. For example, if the target is flying on any radial from the Z axis and away from the weapon system, φ would be that for the rear view of the target ($\varphi = \pi$) and θ would progress from $\theta = 0$ when the target is directly over the weapon site to $\theta = \pi/2$ as it approaches the horizon.

FIGURE 2.35-1. Azimuth Interpolation.

FIGURE 2.35-2. Elevation Interpolation.

The ellipsoidal width to height ratio for a given aspect angle can be interpolated from the ellipsoidal width to height ratios of the three relevant views of 6-view data (left/right, top/bottom, front/rear). Similarly, the target area presented to the weapon system can be interpolated from the relevant areas. The following interpolation schemes for 6- and 26-view presented area data are used.

6-View Presented Area Data Interpolation

Consider the azimuth angle φ between the x axis (pointing north) and the target position vector (vector between weapon and target locations) projected on the horizontal x-y plane in the weapon frame; the angle θ is between the z axis (vertical) and the target position vector; θ is the complementary angle of the target elevation angle from the weapon system. An azimuth interpolation factor, φ_f, is calculated (based on φ) to scale the target horizontal contribution to the total target area presented to the weapon system; an elevation interpolation factor, θ_f, is calculated (based on θ) to scale the target vertical contribution to the total presented area. A total of three cross-sectional areas (A_1, A_2, A_3) will be used to determine the total target area presented to the weapon system (one contribution from each of the three on-axis views). The area usage will be explained in the paragraphs which follows. The factors are needed for the presented area algorithm specified by Equation [2.35-30].

If φ is greater than $\pi/2$ and less than $3\pi/2$, the rear view of the aircraft is visible from the weapon system. In this case, the ellipsoidal ratio for the rear view of the target is the first ratio, M_1, for use in the interpolation; the presented area of the rear view is used as the first area, A_1, for calculating the total cross-sectional area of the target presented to the weapon system (defined by Equation [2.35-31]). For this range of φ, φ_f is the ratio of the absolute difference between the true target aspect (φ) and the straight rear view (π) to the maximum deviation from the rear view ($\pi/2$). The azimuth interpolation data for this case are as follows:

$$\varphi_f = \frac{|\varphi - \pi|}{0.5\pi} \qquad [2.35\text{-}14]$$

$$\varepsilon_1 = \varepsilon_B \qquad [2.35\text{-}15]$$

$$A_1 = A_B \quad \text{if} \quad \frac{\pi}{2} < \varphi < \frac{3\pi}{2} \qquad [2.35\text{-}16]$$

where: φ = Target azimuth angle from weapon system (rad)
ε_B = Ellipsoidal ratio of rear view of target
A_B = Presented area of rear view of target (m^2)

If φ does not fall between $\pi/2$ and $3\pi/2$, the front view of the aircraft is visible from the weapon system. In this case, the ellipsoidal ratio for the front view of the target is the first ratio (ε_1) for use in the interpolation; the presented area of the front view is used as the first area (A_1) for calculating the total cross-sectional area of the target presented to the weapon system. The azimuth interpolation data for this case are as follows:

$$\varphi_F = \begin{cases} \dfrac{(2\pi - \varphi)}{0.5\pi} & \text{if } \dfrac{3\pi}{2} \le \varphi < 2\pi \\ \dfrac{\varphi}{0.5\pi} & \text{if } 0 \le \varphi \le \dfrac{\pi}{2} \end{cases} \quad [2.35\text{-}17]$$

$$\varepsilon_1 = \varepsilon_F \quad [2.35\text{-}18]$$

$$A_1 = A_F \text{ if } 0 \le \varphi \le \frac{\pi}{2} \text{ or } \frac{3\pi}{2} \le \varphi < 2\pi \quad [2.35\text{-}19]$$

where: φ = Target azimuth angle from weapon system (rad)
ε_F = Ellipsoidal ratio of front view of target
A_F = Presented area of front view of target (m^2)

If φ is less than π, the left side of the aircraft is visible. In this case, the ellipsoidal ratio for the left side of the target is the second ratio, ε_2, for use in the interpolation; the presented area of the left view is used as the second area, A_2, for calculating the total cross-sectional area of the target presented to the weapon system:

$$\varepsilon_2 = \varepsilon_L \quad [2.35\text{-}20]$$

$$A_2 = A_L \text{ if } 0 \le \varphi < \pi \quad [2.35\text{-}21]$$

where: ε_L = Ellipsoidal ratio of left side of target
A_L = Presented area of left side of target (m^2)

If φ is equal to or greater than π, the right side of the aircraft is visible from the weapon system. In this case, the ellipsoidal ratio for the right side of target is the second ratio (ε_2) for use in the interpolation; the presented area of the right view is used as the second area (A_2) for calculating the total cross-sectional area of the target presented to the weapon system:

$$\varepsilon_2 = \varepsilon_R \quad [2.35\text{-}22]$$

$$A_2 = A_R \text{ if } \pi \le \varphi < 2\pi \quad [2.35\text{-}23]$$

where: ε_R = Ellipsoidal ratio of right side of target
A_R = Presented area of right side of target (m^2)

If θ is less than or equal to $\pi/2$, part of the bottom of the target is exposed to the weapon system. In this case, the ellipsoidal ratio calculated for the bottom of the target is the third ratio, ε_3, for use in the interpolation. The elevation interpolation factor, θ_f, is the ratio of the difference between the target aspect and the bottom view (θ - 0 deg) to the maximum deviation from the bottom view ($\pi/2$). For this range of θ values, the presented area of the bottom view is used as the third area, A_3, of the three total used for calculating the total

cross-sectional area of the target presented to the weapon system. θ_f is calculated for this case as follows:

$$\theta_f = \frac{\theta}{0.5\pi} \quad [2.35\text{-}24]$$

$$\varepsilon_3 = \varepsilon_U \quad [2.35\text{-}25]$$

$$A_3 = A_U \text{ if } 0 \le \theta \le \frac{\pi}{2} \quad [2.35\text{-}26]$$

where: θ = Elevation aspect angle of target (rad)
ε_U = Ellipsoidal ratio of bottom view of target
A_U = Presented area of bottom view of target (m^2)

If θ is greater than $\pi/2$, part of the top of the target is exposed to the weapon system. In this case, the ellipsoidal ratio for the top of the target is the third ratio (ε_3) for use in the interpolation. For this range of θ values, the presented area of the top view is used as the third area (A_3) of the three total used for calculating the total cross-sectional area of the target presented to the weapon system.

The elevation interpolation factor is the difference between the top view and the true aspect angle over the maximum deviation from the top view:

$$\theta_f = \left| \frac{\theta}{0.5\pi} - 2.0 \right| \quad [2.35\text{-}27]$$

$$\varepsilon_3 = \varepsilon_T \quad [2.35\text{-}28]$$

$$A_3 = A_T \text{ if } \frac{\pi}{2} < \theta \le \pi \quad [2.35\text{-}29]$$

where: θ = Elevation aspect angle of target (rad)
ε_T = Ellipsoidal ratio of top view of target
A_T = Presented area of top view of target (m^2)

The constant 2.0 in Equation [2.35-27] adjusts the elevation interpolation factor so that for values of θ from $\pi/2$ to π the elevation factor progresses from 1 to 0 in a manner similar to the bottom view. Pure top or bottom views generate an elevation interpolation factor of 0 while wings level views generate a value of 1.

The interpolation between the three ellipsoidal ratios results in the ellipsoidal ratio, ε, of the target area presented to the weapon system.

$$\varepsilon = \theta_f(\varphi_f \varepsilon_2 + (1 - \varphi_f)\varepsilon_1) + (1 - \theta_f)\varepsilon_3 \quad [2.35\text{-}30]$$

The total target cross-sectional area, A_t, presented to the weapon system and round is found using the following algorithm:

$$A_t = \frac{A_1 V_{rx} + A_2 V_{ry} + A_3 V_{rz}}{\sqrt{V_{rx}^2 + V_{ry}^2 + V_{rz}^2}} \qquad [2.35\text{-}31]$$

where: A_1, A_2, A_3 = Areas of each of three on-axis views of target presented to weapon (m^2)

V_{rx}, V_{ry}, V_{rz} = Components of round velocity relative to target frame (m/s)

26-View Presented Area Data Interpolation

Target area interpolation for a 26-view presented area data format is accomplished with a different algorithm than that of the 6-view format. In the 26-view case, the presented area data are given at 45-degree azimuthal increments (8 views in 360 degrees) for elevation aspects of zero, 45, and -45 degrees; these account for 24 views. The top and bottom views, (elevation aspect angles of -90˚ and 90˚) result in a total of 26 views. The Computation of Vulnerable Area and Repair Time (COVART) simulation definitions for 26-view presented area data are used in RADGUNS. Figure 2.35-3 depicts the target frame of reference (in RADGUNS and COVART), and the presented area data available for specific target orientations are provided in Table 2.35-1. The table has numbers 1 through 26 listed in the "View" column; these numbers indicate the order of presented area data in a data set.

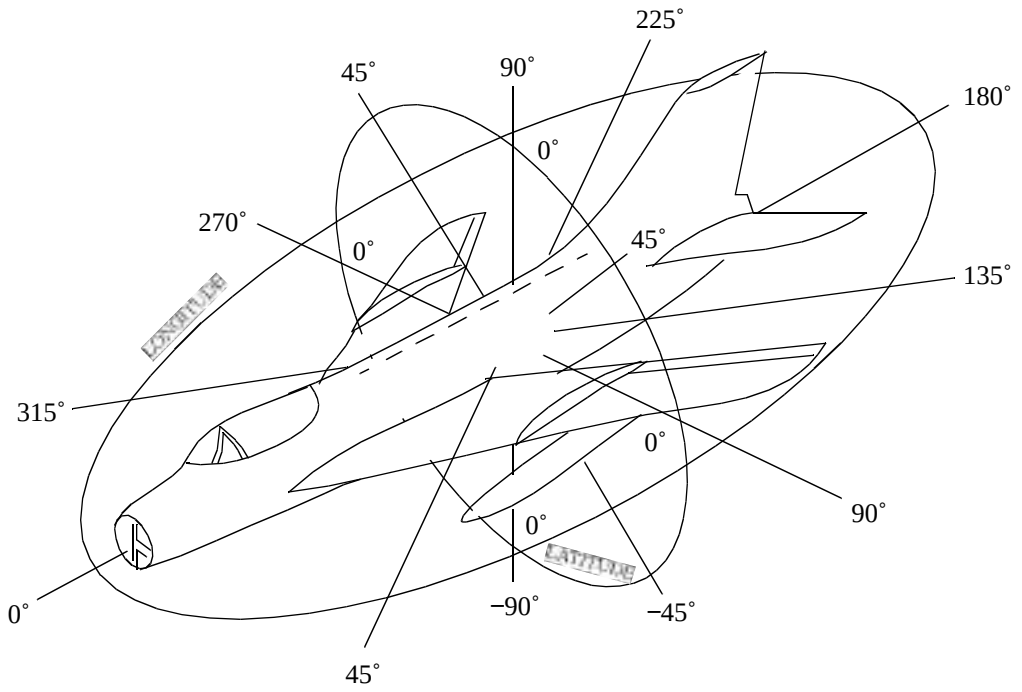

FIGURE 2.35-3. COVART Latitude and Longitude Conventions.

TABLE 2.35-1. 26-View Latitudes and Longitudes.

6 View Equivalent	View	Latitude	Longitude
Bottom	1	-90	90
	2	-45	180
	3	-45	225
	4	-45	270
	5	-45	315
	6	-45	0
	7	-45	45
	8	-45	90
	9	-45	135
Rear	10	0	180
	11	0	225
Right side	12	0	270
	13	0	315
Front	14	0	0
	15	0	45
Left side	16	0	90
	17	0	135
	18	45	180
	19	45	225
	20	45	270
	21	45	315
	22	45	0
	23	45	45
	24	45	90
	25	45	135
Top	26	90	90

Consider the azimuth and elevation aspect angles (φ and θ, respectively) of the target as viewed by the round at the closest point of approach. An azimuth interpolation factor, φ_f, scales the target horizontal contribution to the total target area presented to the round; an elevation interpolation factor, θ_f, scales the target vertical contribution to the total presented area. φ_f and θ_f each is the fraction of the angle exceeding the nearest increment in azimuth and elevation, respectively:

$$\varphi_f = \frac{\varphi - \varphi_L}{45°} \qquad [2.35\text{-}32]$$

where: φ = Target azimuth aspect angle (deg)

φ_L = Angle value below φ which has a presented area data value (deg)

$$\theta_f = \frac{\theta - \theta_L}{45°} \qquad [2.35\text{-}33]$$

where: θ = Target elevation aspect angle (deg)
θ_L = Angle value below θ which has a presented area data value (deg)

A total of four cross-sectional areas generally are used to determine the total target area presented to the round; the use of elevation aspect angle data at 90 or -90 degrees results in no azimuth interpolation at one elevation bound, and only three cross-sectional areas are the subject of interpolation. The target view from the projectile has presented area data above and below the elevation aspect angle. Similarly, presented area data are at points in azimuth smaller and larger than the value of the azimuthal aspect viewed by the round. The area presented to the round is found by linear interpolation between elevation angles at each of the two azimuth values, followed by linear interpolation between the interpolated elevation values found at each of the two azimuth data points; this method is described by the following three equations.

The lower azimuth value has a presented area, $A_{\varphi 1}$, interpolated from the areas above and below the azimuth:

$$A_{\varphi 1} = A_1 + (A_2 - A_1)\theta_f \qquad [2.35\text{-}34]$$

where: A_1, A_2 = Presented area values at elevation angles surrounding lower azimuth angle (m^2)
θ_f = Elevation interpolation factor defined by Equation [2.35-33]

The upper azimuth value has a presented area, $A_{\varphi 2}$, interpolated as follows:

$$A_{\varphi 2} = A_3 + (A_4 - A_3)\theta_f \qquad [2.35\text{-}35]$$

where: A_3, A_4 = Presented area values at elevation angles surrounding lower azimuth angle (m^2)
θ_f = Elevation interpolation factor defined by Equation [2.35-33]

Finally, the presented area of the target to the round, A, is found through azimuthal interpolation:

$$A = A_{\varphi 1} + (A_{\varphi 2} - A_{\varphi 1})\varphi_f \qquad [2.35\text{-}36]$$

where: $A_{\varphi 1}$, $A_{\varphi 2}$ = Presented area values at elevation angles surrounding lower azimuth angle (m^2)
φ_f = Azimuthal interpolation factor defined by Equation [2.35-32]

Design Element 35-4: Diffuse Target Definition

The P_h depends on the two-dimensional probability densities of the target and round in the plane perpendicular to the round velocity and passing through the target center (Reference

18, page 3). The target area distribution is assumed to be elliptic Gaussian with variances (S_x^2, S_y^2) (see Section 2.35.4). Thus, the probability density of a target located at (x, y) is (Reference 18, page 2):

$$P_T(x, y) = \exp\left[-\frac{1}{2}\left(\frac{x^2}{S_x^2} + \frac{y^2}{S_y^2}\right)\right] \quad [2.35\text{-}37]$$

which when integrated over the entire real field yields the diffuse target presented area, A_D:

$$A_D = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} P_T(x, y)dxdy = 2\pi S_x S_y \quad [2.35\text{-}38]$$

where: S_x = Standard deviation of target area along x axis (m)
S_y = Standard deviation of target area along y axis (m)

The target ellipsoidal ratio and area described by Equations [2.35-30 and 2.35-31] can be used to determine the standard deviation of the diffuse target presented area. Using $S_x = \varepsilon S_y$, equating A_D to A_t and rearranging factors of Equation [2.35-38] yields the standard deviations of the target presented area:

$$S_y = \sqrt{\frac{\varepsilon A_t}{2\pi}} \quad [2.35\text{-}39]$$

$$S_y = \sqrt{\frac{A_t}{2\pi\varepsilon}} \quad [2.35\text{-}40]$$

where: A_t = Equivalent diffuse target presented area (m^2)
ε = Ellipsoidal ratio of the target presented area

Design Element 35-5: Dispersion of Rounds

Multiple rounds in a burst have slightly different trajectories. In general, the rounds diverge and can be characterized by an angular separation (angular dispersion), ß. The angular dispersion of the rounds determines their separation distance at the range of the rounds at the CPA between the target centroid and the center of the distribution of rounds. Round separation is measured with respect to the plane normal to the position vector (from the weapon system) of the center of a burst and passing through the target centroid; this right triangle geometry allows the use of the Pythagorean Theorem to obtain the standard deviation, σ_B, of the distribution of burst rounds in the plane:

$$\sigma_B = R \tan \beta \quad [2.35\text{-}41]$$

where: R = Range from weapon to the center of the burst at CPA (m)
ß = Angular dispersion of rounds in a burst (rad)

Design Element 35-6: Single Round Probability of Hit

The scenario geometry assumptions in Section 2.35.4 require the P_H to be determined using a plane perpendicular to the center of the salvo velocity vector; the plane's cartesian origin is located at the target centroid. The target area distribution, P_T, is assumed to be elliptic Gaussian with variances (S_x^2, S_y^2), and is described by Equation [2.35-37]. Another assumption is that the actual location of a round on the plane is distributed as bivariate independent Gaussian, with means (X, Y), and variances (σ_x^2, σ_y^2) (Reference 18, page 1). Thus, the probability density function for a round, P_R, located at (x, y) (Reference 18, page 23) is:

$$P_R(x, y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left[-\frac{1}{2}\left(\frac{(x-X)^2}{\sigma_x^2} + \frac{(y-Y)^2}{\sigma_y^2}\right)\right] \qquad [2.35\text{-}42]$$

The P_H, of a round at point (x, y) hitting the target is the product of the target and round probability densities since they are assumed to be independent (Reference 21, Section 2.5):

$$P_H(x, y) = P_T(x, y)P_R(x, y) \qquad [2.35\text{-}43]$$

Thus, the probability of hitting the target is:

$$P_H = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} P_T(x, y)P_R(x, y)dxdy$$

$$= \frac{1}{2\pi\sigma_x\sigma_y}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \exp\left[-\frac{1}{2}\left(\frac{x^2}{S_x^2} + \frac{y^2}{S_y^2}\right)\right]\exp\left[-\frac{1}{2}\left(\frac{(x-X)^2}{\sigma_x^2} + \frac{(y-Y)^2}{\sigma_y^2}\right)\right]dxdy \qquad [2.35\text{-}44]$$

A complete solution to the integral of Equation (2.35-44) is contained in Reference 18, pages 23 through 28. The resulting equation for single round P_h is:

$$P_H = \frac{S_x S_y}{(\sigma_x^2 + S_x^2)^{\frac{1}{2}}(\sigma_y^2 + S_y^2)^{\frac{1}{2}}} \exp\left[-\frac{1}{2}\left(\frac{X^2}{\sigma_x^2 + S_x^2} + \frac{Y^2}{\sigma_y^2 + S_y^2}\right)\right] \qquad [2.35\text{-}45]$$

where: X,Y = Mean coordinate values of the salvo center on the plane entered at target (m)

S_x,S_y = Standard deviation of target area distribution (m)

σ_x,σ_y = Standard deviation of rounds on plane perpendicular to salvo (m)

Design Element 35-7: Probability of Hit for a Burst

Several rounds can be fired in a single burst. The P_h for a given number of randomly dispersed rounds fired simultaneously at a single aim point (a burst) is described by “the Salvo Formula” derived in Reference 19. An assumption of the formula is that a bivariate Gaussian distribution of possible aim points is caused by random tracking and aiming errors rather than a discrete aim point error for each projectile fired. The aim point error (distance), D, is the separation between the target centroid and the AAA system-predicted

target position. The geometry and definitions applicable to the Salvo Formula are illustrated below in Figure 2.35-4, which is Figure C-1 of Reference 20. The methodology used to determine the Salvo Formula is described in detail in Reference 19; the formula for the P_h for a burst is as follows:

$$P_{h_{BUR}} = \sum_{i=1}^{N} \binom{N}{i} (-1)^{i+1} \left(\frac{A}{2\pi\sigma_b^2 + A}\right)^{i-1} \left(\frac{A}{2\pi(i\sigma_a^2 + \sigma_b^2) + A}\right) \exp\left(\frac{-i\pi D^2}{2\pi(i\sigma_a^2 + \sigma_b^2) + A}\right) \quad [2.35\text{-}46]$$

where:
- N = Number of rounds in burst
- A = Average area of target (m^2)
- D = Average aim point error (m)
- σ_a = Standard deviation of D (m)
- σ_b = Standard deviation of rounds (m)

The first factor in the summation can be solved iteratively on a computer by representing it at the i^{th} iteration in terms of the previous iteration.

The number of different combinations that can be formed from *N* distinct objects taken *i* at a time is (Reference 21, page 523):

$$\binom{N}{i} = \frac{N!}{i!(N-1)!} = \frac{N!}{i(i-1)!(N-i)!} \quad [2.35\text{-}47]$$

The combination of *N* things taken *(i-1)* at a time is:

$$\binom{N}{i-1} = \frac{N!}{(i-1)!(N-(i-1))!} = \frac{N!}{(i-1)!(N-(i-1))(N-i)!} \quad [2.35\text{-}48]$$

Dividing Equation [2.35-47] by Equation [2.35-48] and rearranging terms yields:

$$\binom{N}{i} = \binom{N}{i-1}\left(\frac{N-(i-1)}{i}\right) \quad [2.35\text{-}49]$$

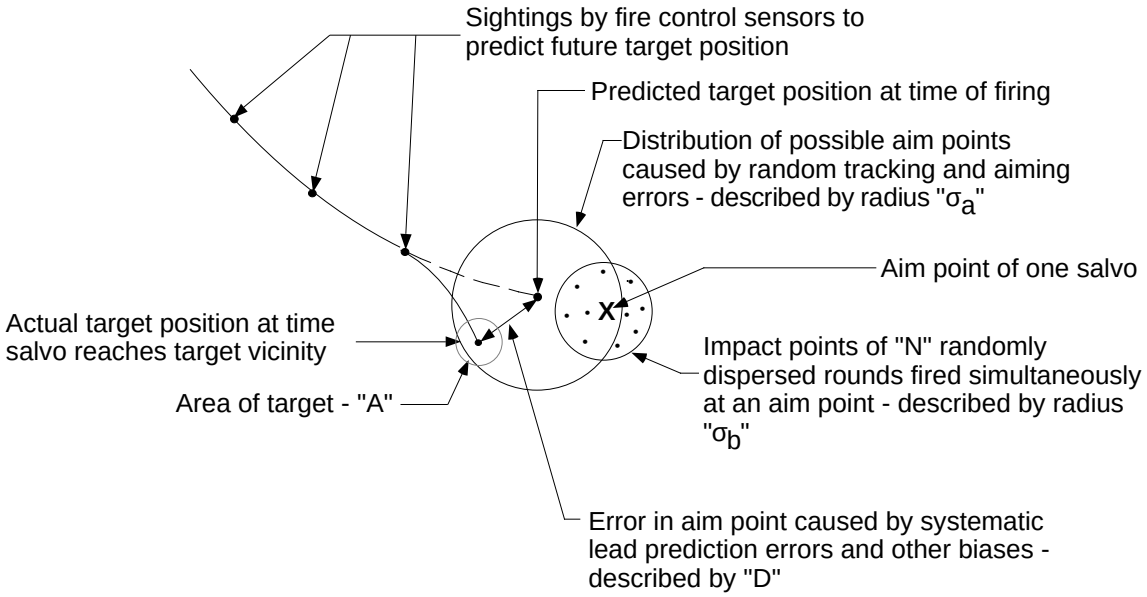

FIGURE 2.35-4. Salvo Formula Geometry and Definitions.

Design Element 35-8: Cumulative Probability of Hit

The P_h for an entire engagement considers the P_h for all rounds fired during an engagement. Each round has an associated aim point, trajectory, and resulting miss distance that contribute to a single shot P_h value. As mentioned in Section 2.35.4, the P_h for each round in a multiple round engagement is assumed to be independent of probabilities of hit of the other rounds. If the probability of each event in a series of *n* events is independent, the probability that all *n* events will happen simultaneously is the product of each of the independent probabilities (Reference 21, Section 2.5):

$$P(1 \cap 2 \cap \quad N) = \prod_{n=1}^{N} P_n \qquad [2.35\text{-}50]$$

Given the probability of a single hit, P_h, the probability of miss, P_m, is:

$$P_m = 1 - P_h \qquad [2.35\text{-}51]$$

The probability that all rounds have missed the target is calculated using Equation [2.35-50]:

$$P_M = \prod_{n=1}^{N} P_m \qquad [2.35\text{-}52]$$

$$= \prod_{n=1}^{N} (1 - P_h)$$

Therefore, the cumulative probability that *at least* one round has hit the target is:

$$P_{hcum} = 1 - P_M \qquad [2.35\text{-}53]$$

Design Element 35-9: Standard Deviation

An important measure of variability of a sample (variable) is the variance. This widely used measure in statistics is denoted as S^2. Given a population of n samples, the variance is expressed as (Reference 21, page 65):

$$S^2 = \frac{n \sum_{i=1}^{n} X_i^2 - \left(\sum_{i=1}^{n} X_i \right)^2}{n(n-1)} \qquad [2.35\text{-}54]$$

The standard deviation, σ, is defined as the square root of the variance (Reference 21, page 65):

$$\sigma = \sqrt{S^2} \qquad [2.35\text{-}55]$$

2.35.3 Functional Element Software Design

This section describes the software design in RADGUNS which implements the Probability of Hit FE requirements and design approach. It is organized as follows: the first subsection describes the subroutine hierarchy and gives descriptions of the relevant subroutines; the next subsection contains logical flow charts and describes important operations represented by each block in the charts; the last subsection contains a description of all input and output data for the functional element as a whole and for the subroutines that implement P_h calculations.

Probability of Hit Subroutine Design

RADGUNS uses Subroutine HITPRB to calculate the P_h. The RADGUNS program main routine is called AAASIM. Figure 2.35-5 shows the call hierarchy associated with the FE. The shaded blocks in the call tree denote the modules that directly implement the FE. Table 2.35-2 contains a brief description of each of these subroutines.

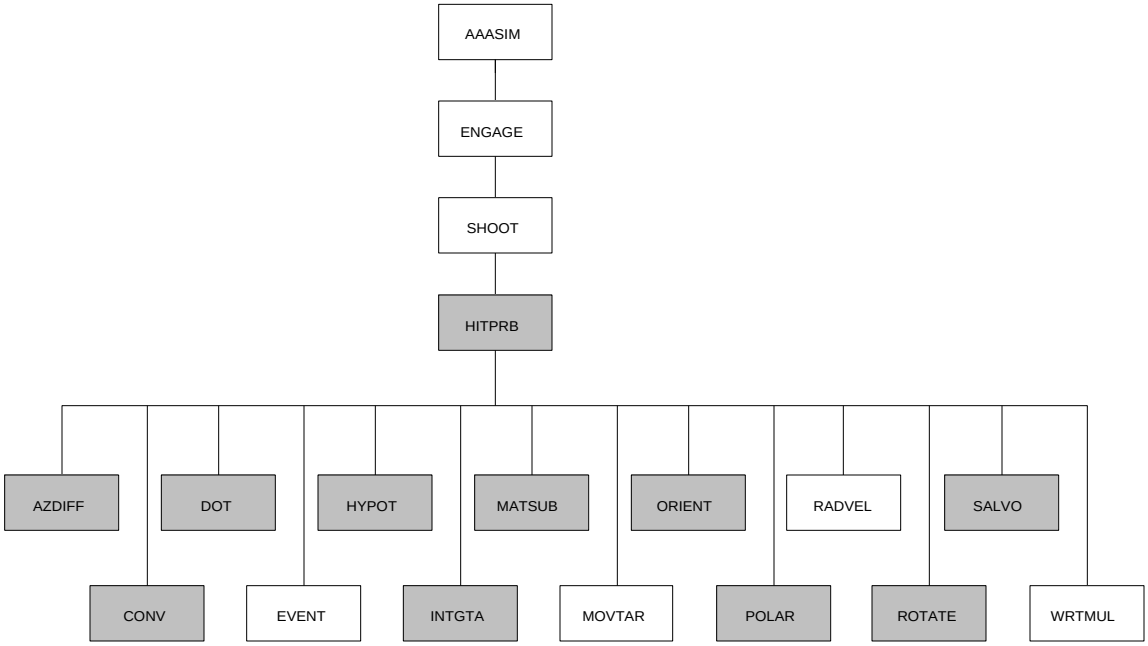

FIGURE 2.35-5. Probability of Hit FE Subroutine Call Hierarchy.

Functional Flow Diagrams

The three flow diagrams that follow describe the main routine for P_h calculations (HITPRB) and two routines called by HITPRB to implement the Salvo Formula (SALVO) and 26-view target presented area determination (INTGTA). These three primary modules implement the FE design elements. The other modules listed above in Table 2.35-2 provide mathematical utility procedures to HITPRB or perform tasks related to data recording and graphical output.

Subroutine HITPRB

Figure 2.35-6 depicts the logic flow of Subroutine HITPRB. The flow diagram blocks are numbered for ease of reference in the discussion that follows the diagram.

Subroutine HITPRB is the main module implementing both the Probability of Hit FE and the Probability of Kill FE. The same algorithms often apply to both FEs. Therefore, some P_k information is included in the flow chart; but, those processes do not implement the subject (Probability of Hit) FE. Also, many flow diagram blocks manipulate interim and final results of P_h calculations for various user output options; the blocks that directly apply to the Probability of Hit FE implement a specific equation number or are interim steps which support a specific equation. The block descriptions for HITPRB cite the equations being implemented.

TABLE 2.35-2. Subroutine Descriptions.

MODULE NAME	DESCRIPTION
AAASIM	Main routine to simulate AAA system
AZDIFF	Returns difference between two azimuth angles such that -B < angle < B
CONV	Converts between degrees and radians, and Soviet mils and radians
DOT	Calculates the dot product of two vectors
ENGAGE	Controls system after target acquisition
EVENT	Records important simulation events
HITPRB	Calculates P_h and P_k for current round; updates burst and cumulative probabilities
HYPOT	Calculates the distance from the origin to a point in three dimensional space
INTGTA	Computes target (26-view) presented area for a given azimuth and elevation from the target to a round
MATSUB	Performs matrix subtraction
MOVTAR	Computes the target position, velocity, and acceleration as a function of time
ORIENT	Computes the target angular orientation (yaw, pitch, roll) with respect to the radar coordinate frame as a function of time
POLAR	Converts from rectangular to polar coordinates
RADVEL	Computes the radial velocity of an object with respect to the radar
ROTATE	Transforms a vector to an equivalent vector in a rotated coordinate frame
SALVO	Computes burst P_h via the Salvo Formula
SHOOT	Controls firing of the guns
WRTMUL	Creates tabular data files and P_k-map, expected hits, and burst P_h plot files
Note: The modules implementing the Probability of Hit Functional Element are identified in bold letters	

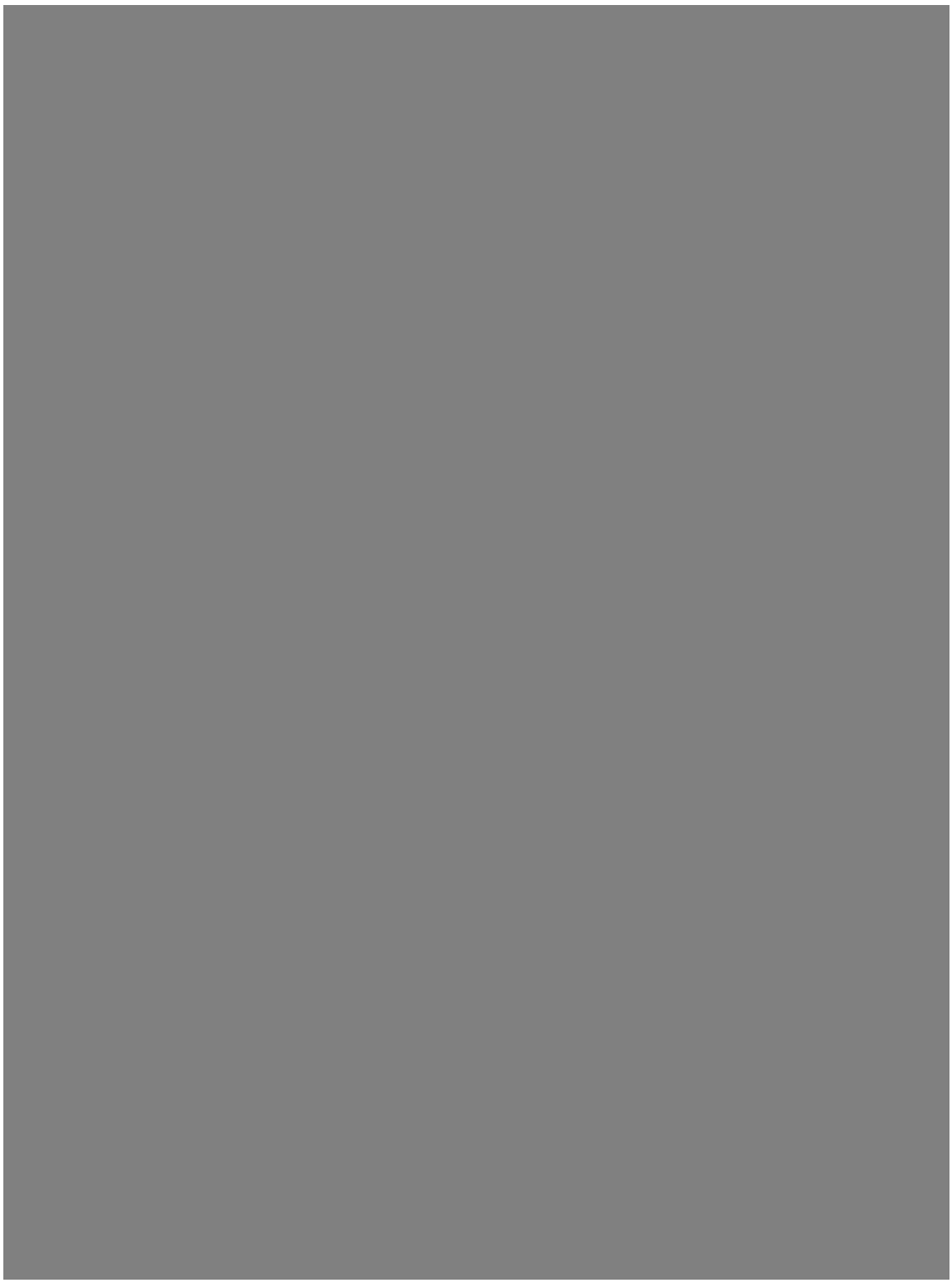

FIGURE 2.35-6. Subroutine HITPRB Flow Diagram.

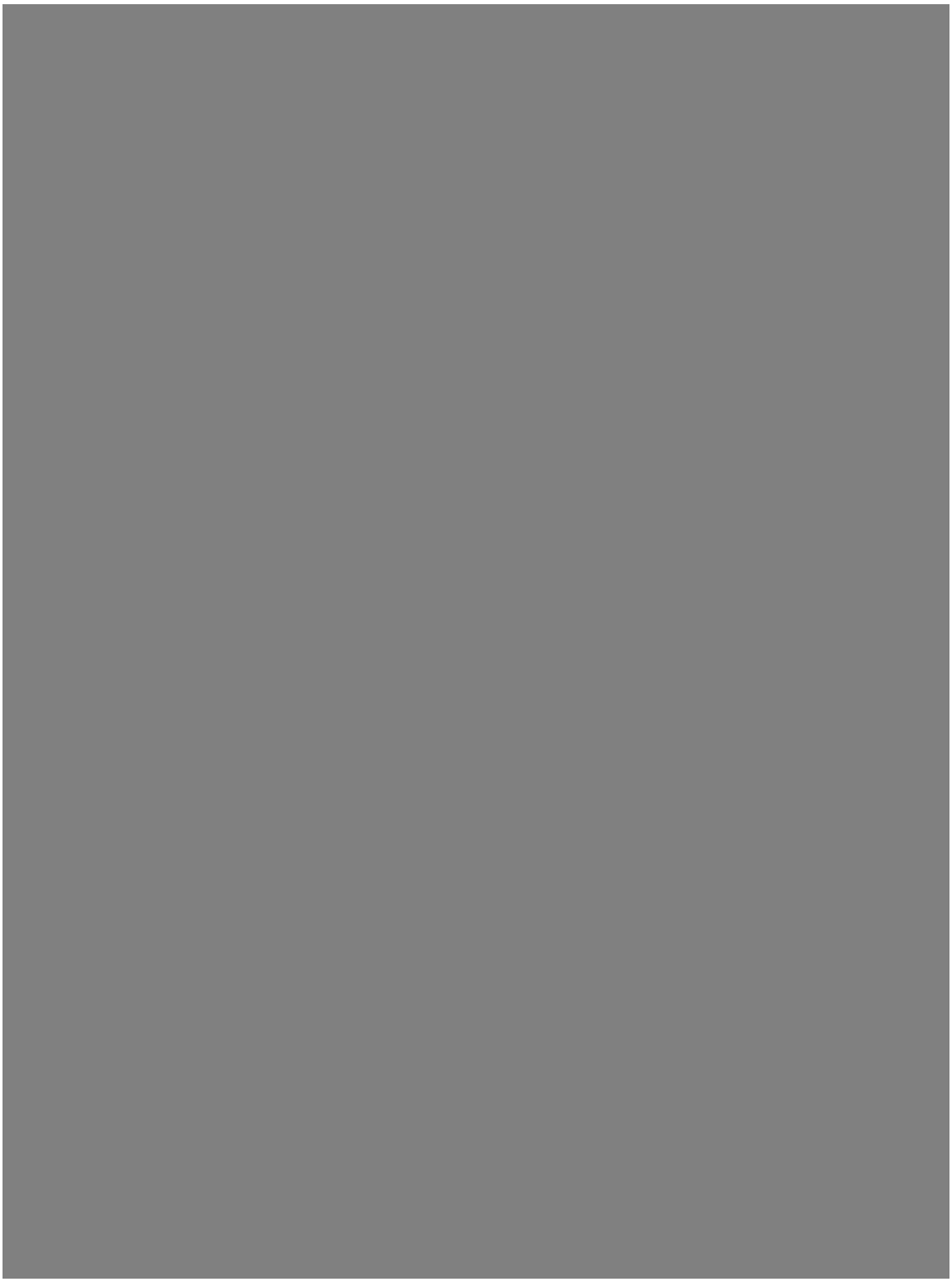

FIGURE 2.35-6. Subroutine HITPRB Flow Diagram. (Contd.)

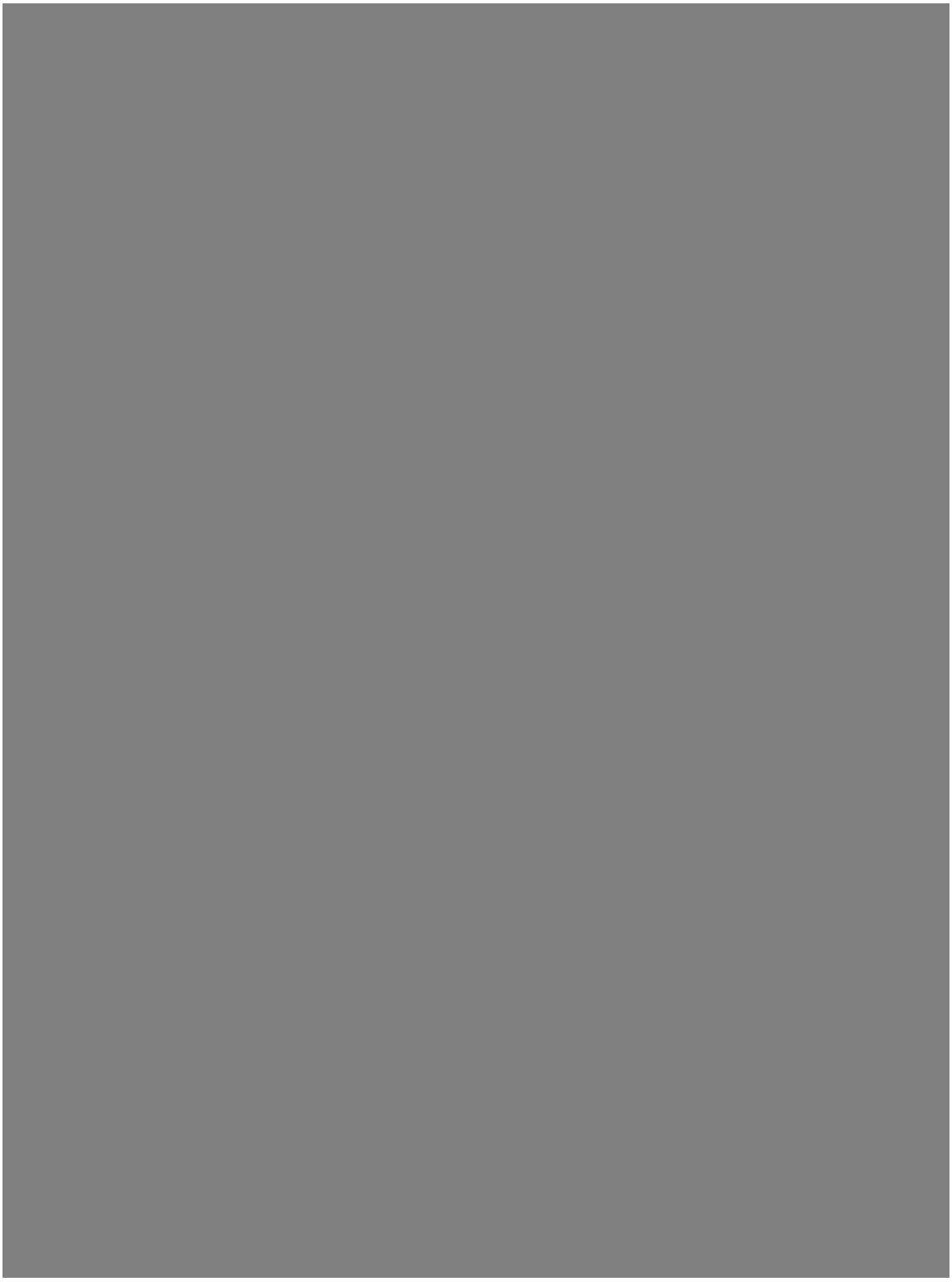

FIGURE 2.35-6. Subroutine HITPRB Flow Diagram. (Contd.)

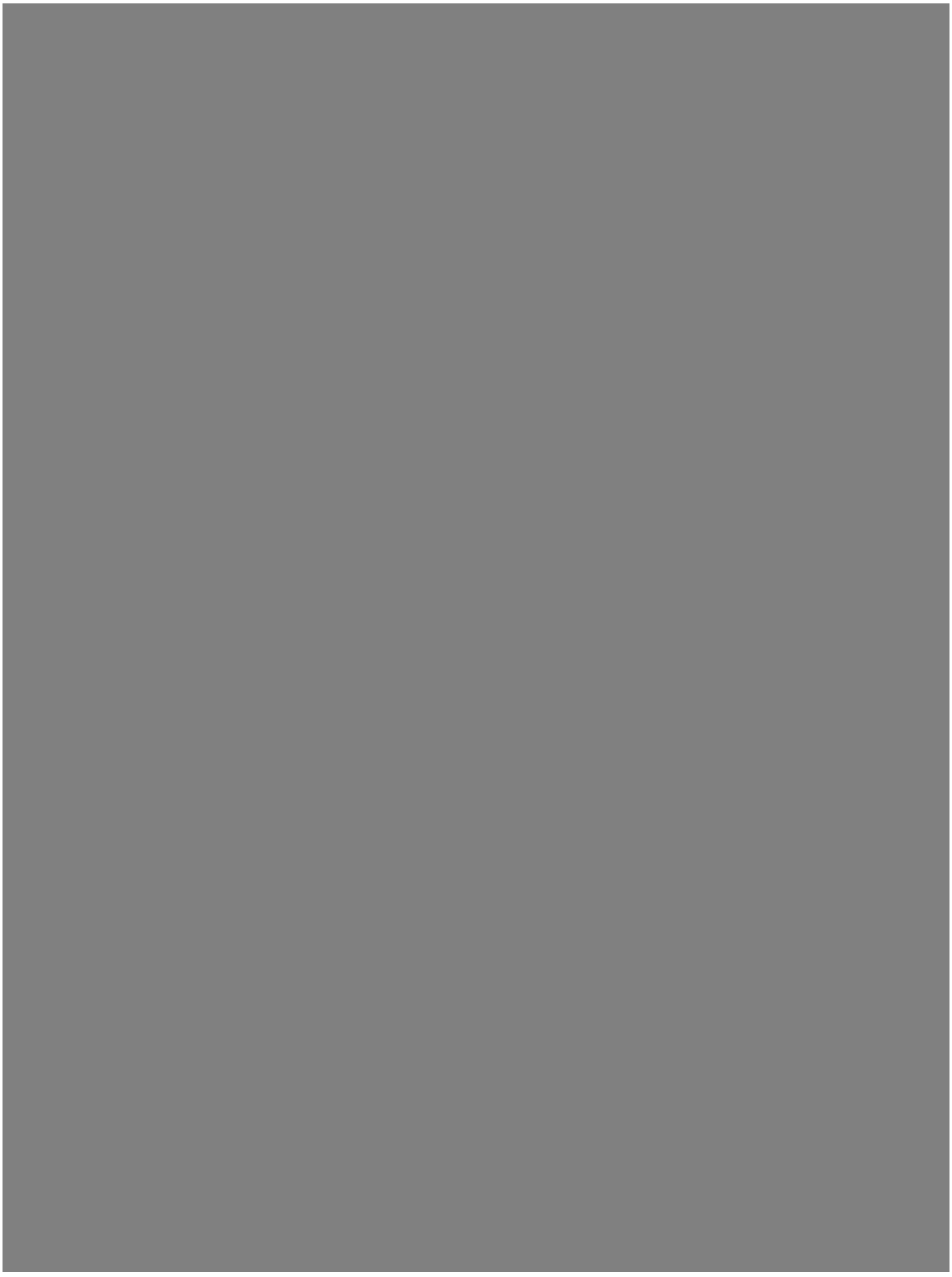

FIGURE 2.35-6. Subroutine HITPRB Flow Diagram. (Contd.)

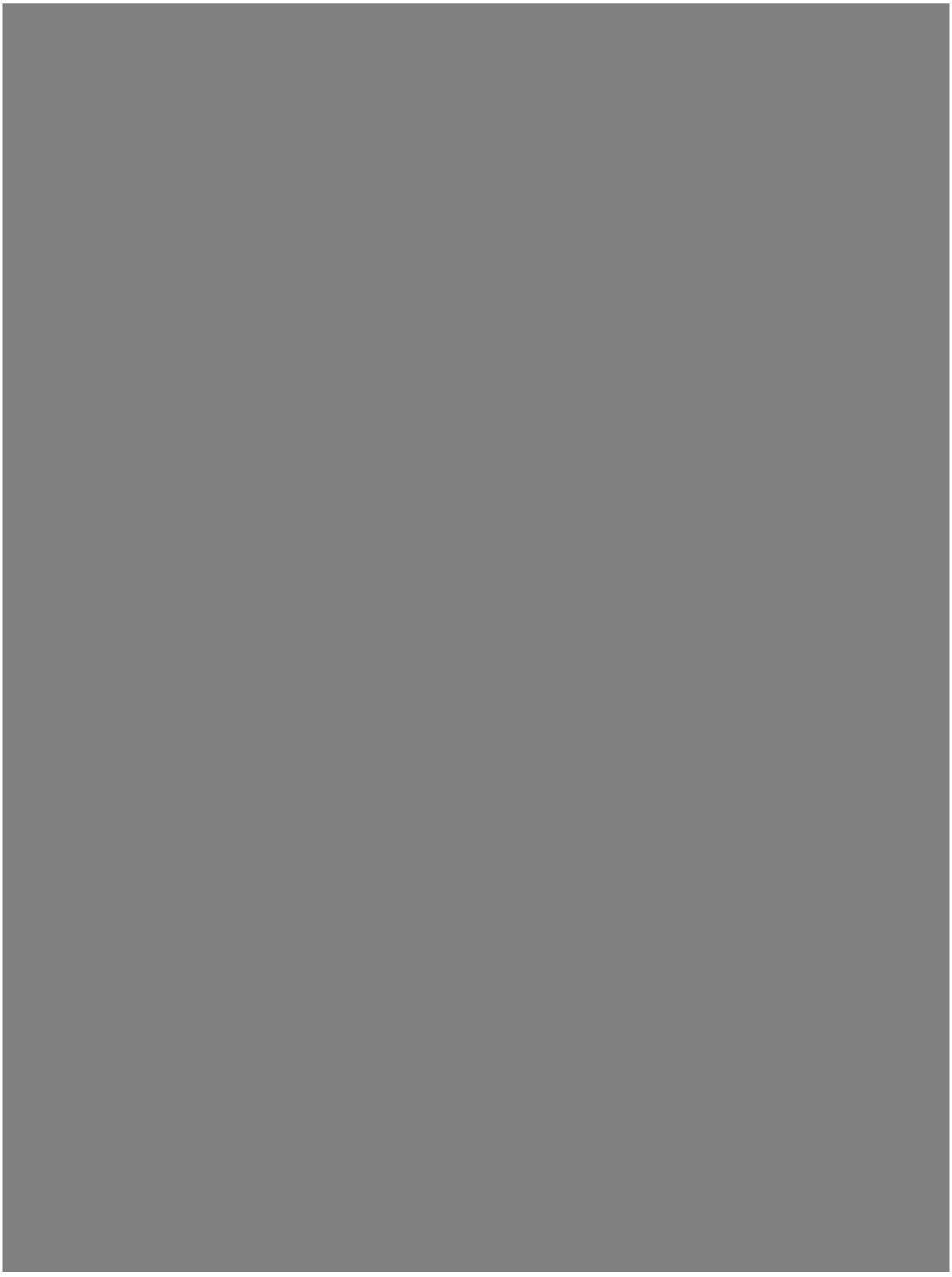

FIGURE 2.35-6. Subroutine HITPRB Flow Diagram. (Contd.)

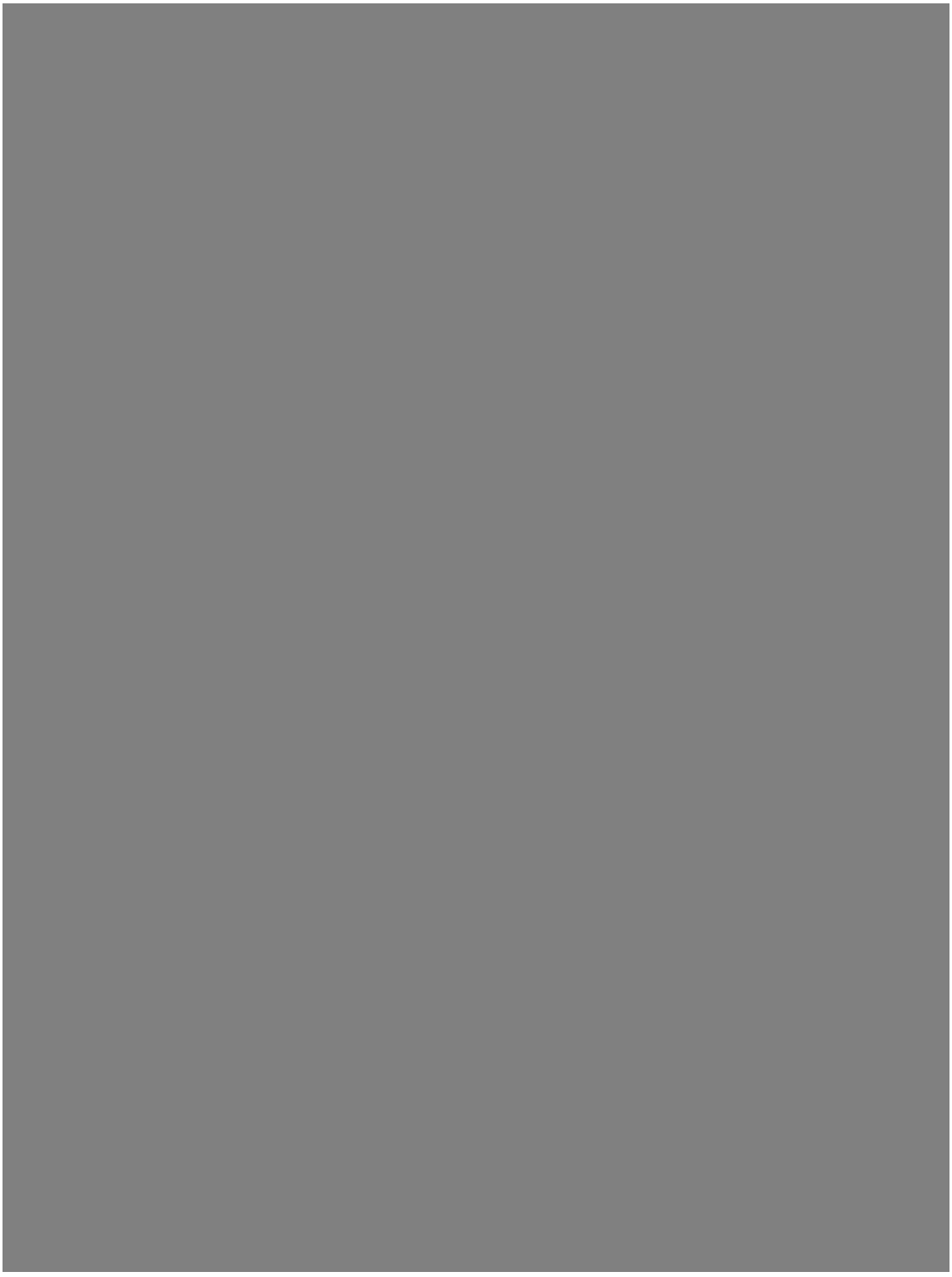

FIGURE 2.35-6. Subroutine HITPRB Flow Diagram. (Contd.)

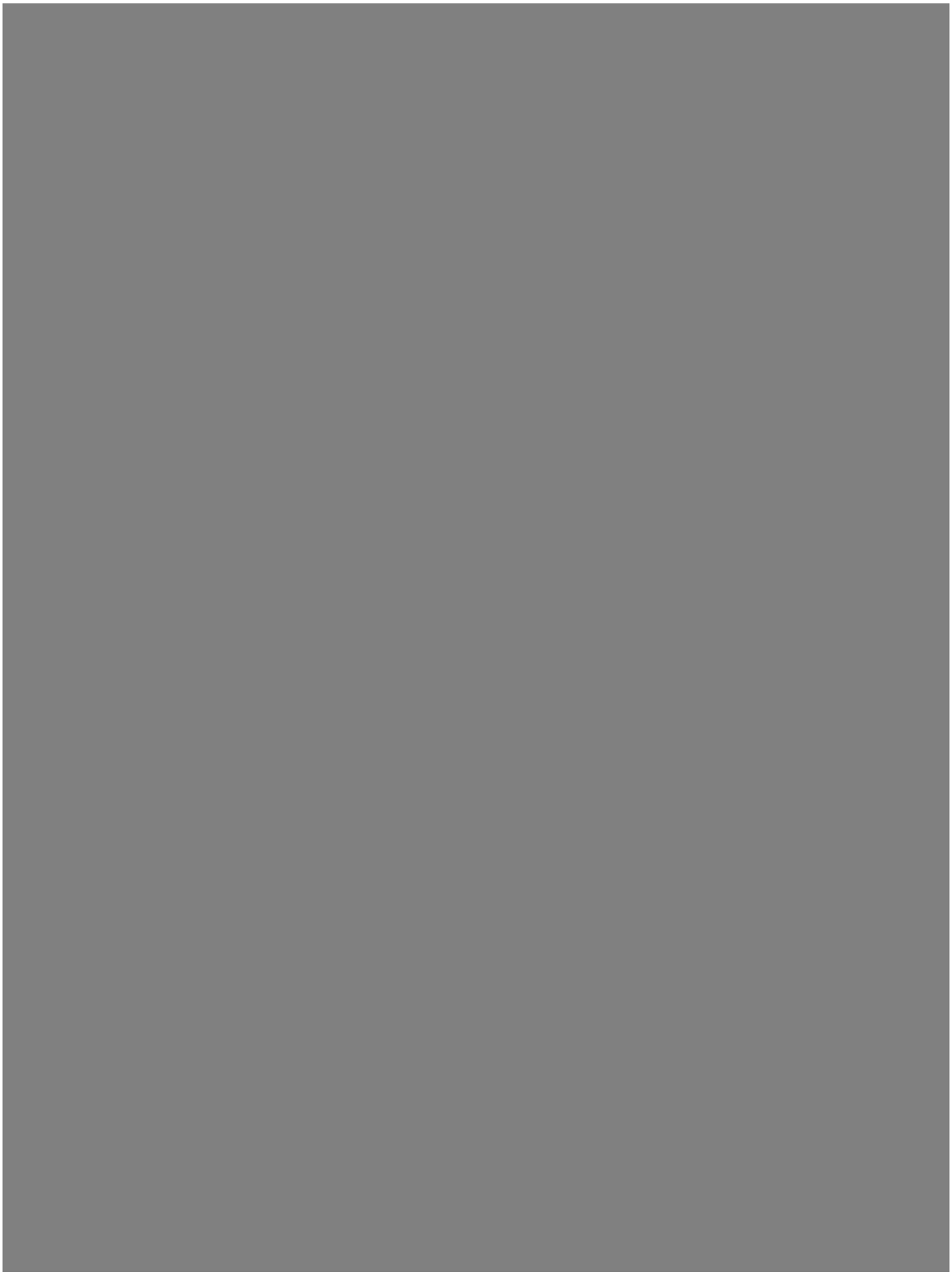

FIGURE 2.35-6. Subroutine HITPRB Flow Diagram. (Contd.)

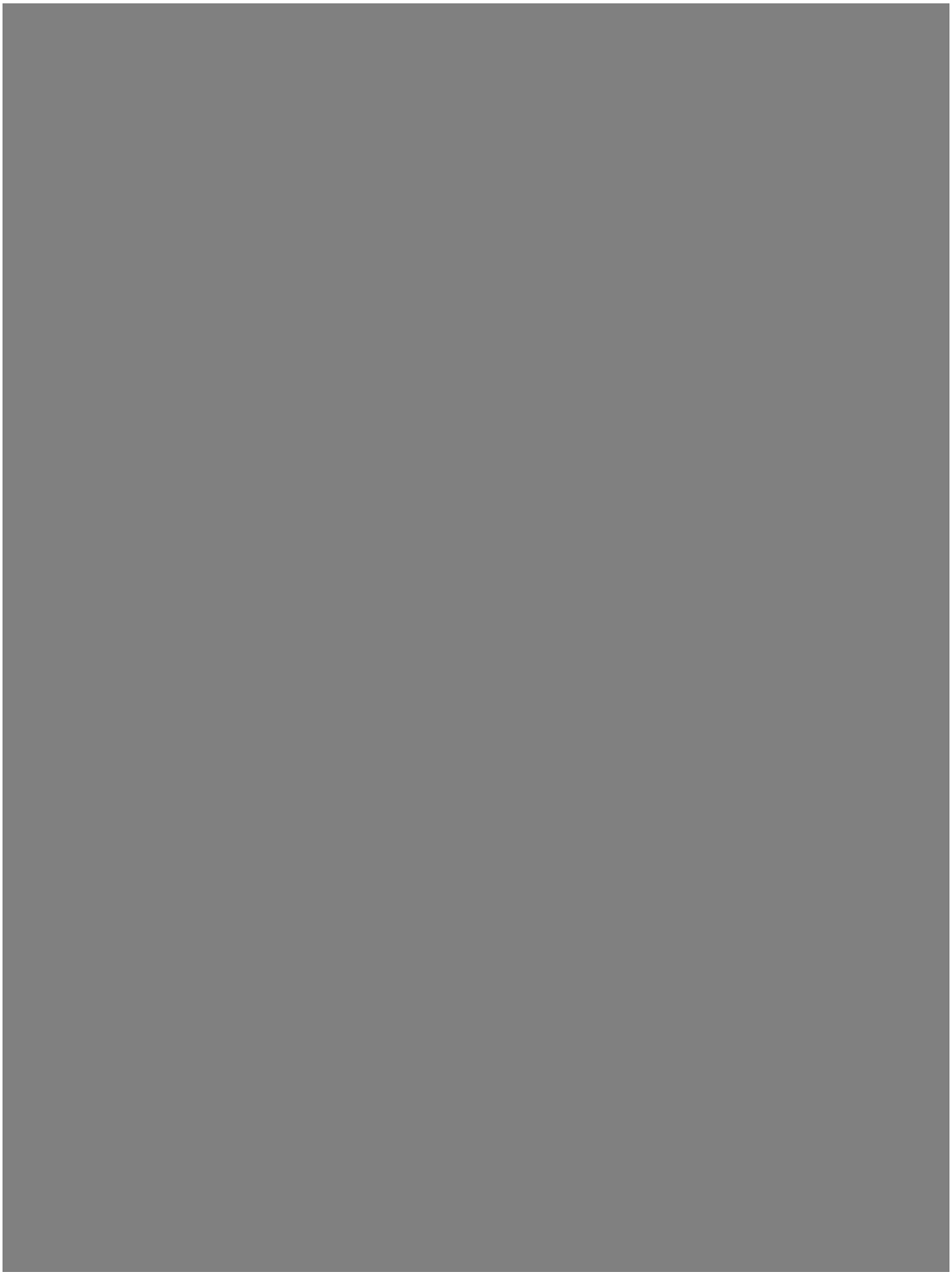

FIGURE 2.35-6. Subroutine HITPRB Flow Diagram. (Contd.)

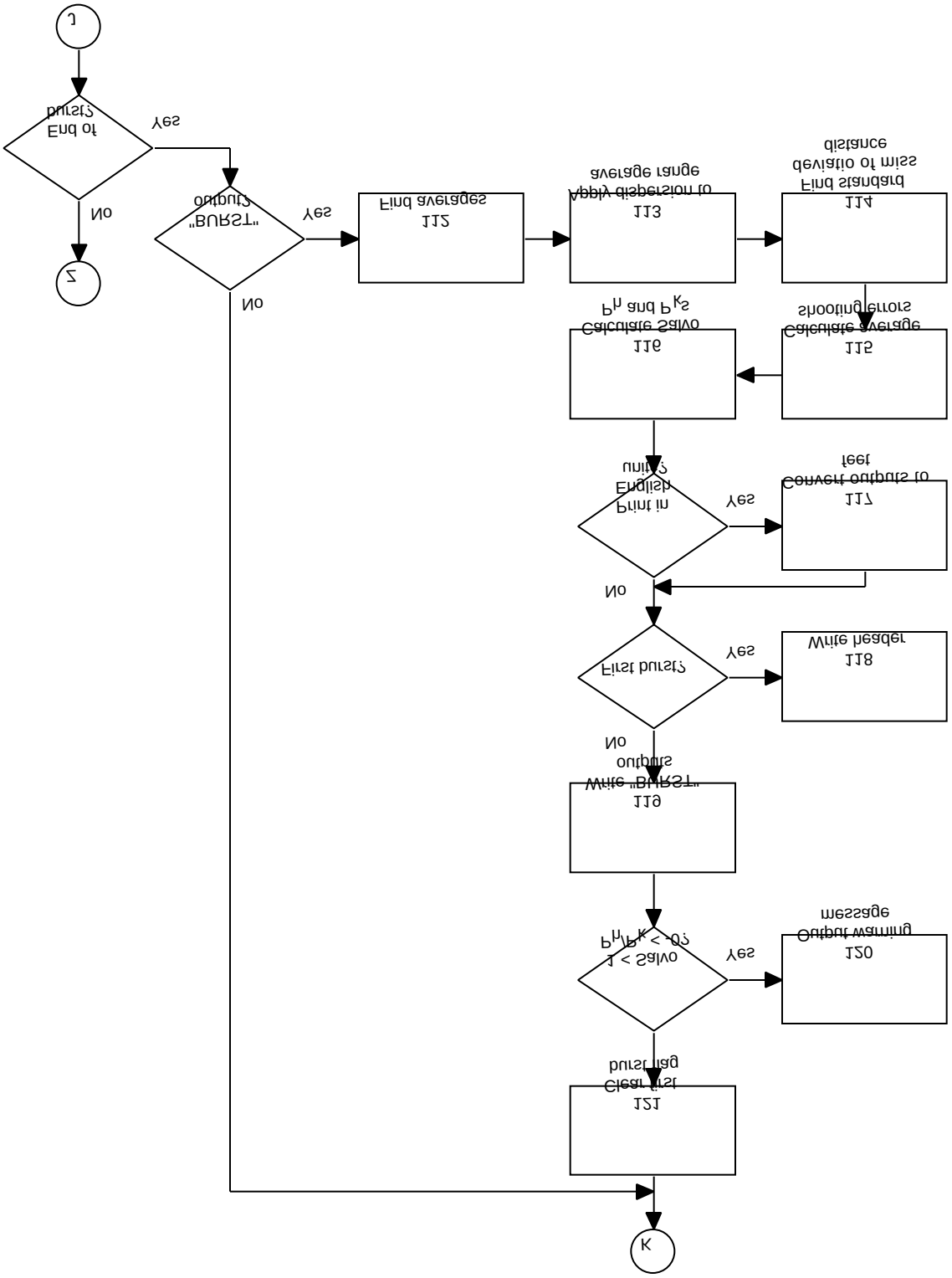

FIGURE 2.35-6. Subroutine HITPRB Flow Diagram. (Contd.)

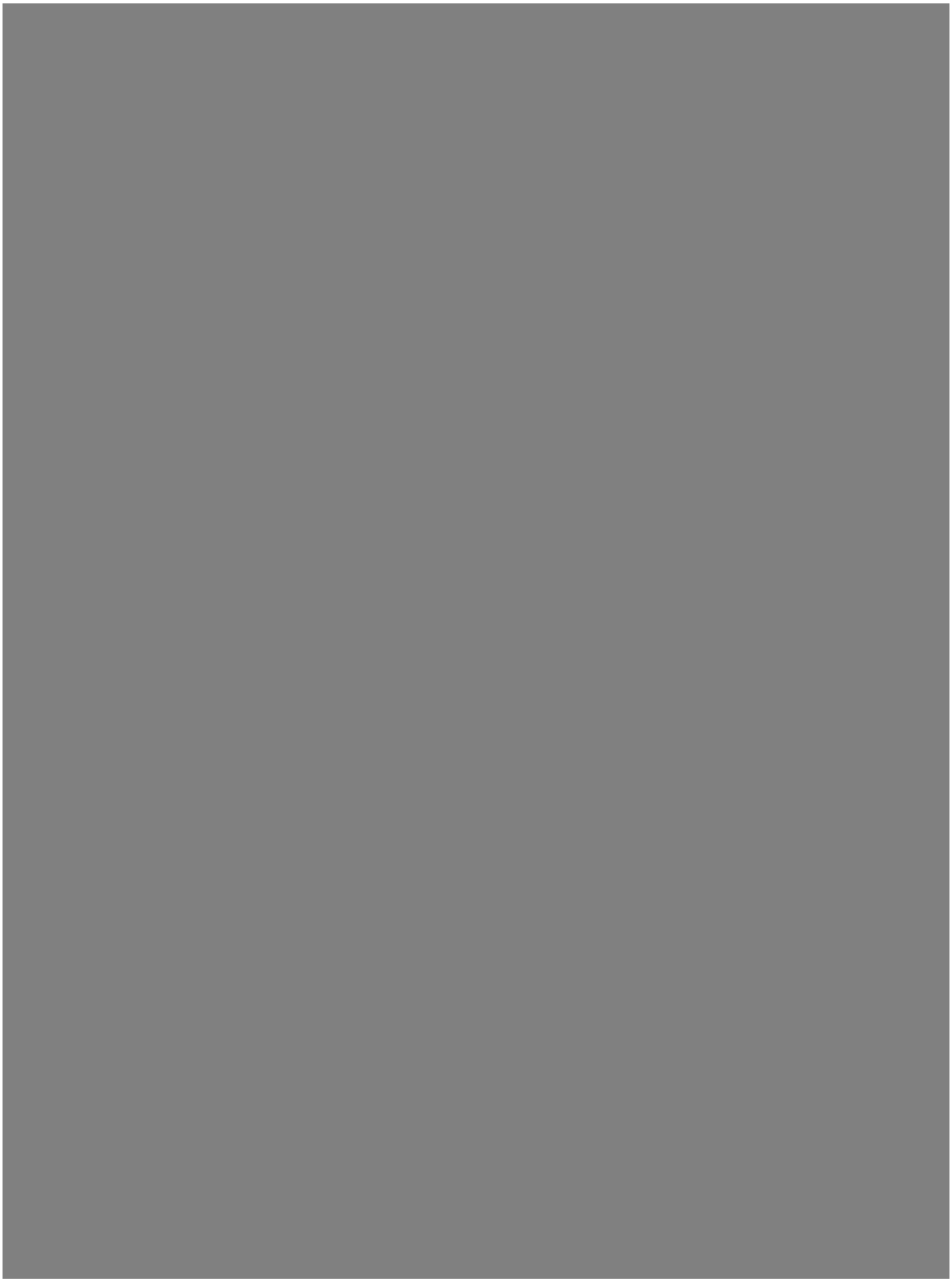

FIGURE 2.35-6. Subroutine HITPRB Flow Diagram. (Contd.)

Block 1: If no rounds are due to be fired this scan, *SBURST*, the start of burst flag, is cleared.

Block 2: If *SBURST* is true, a burst is scheduled to start, and several variables are initialized. The probabilities that all rounds fired to this point will miss the target (both hit and kill) are set to 1.0. The burst probabilities of hit and kill (four types), the summation variables used in the calculation of burst averages for use in the Salvo Equation, the number of rounds fired in the current burst (*NSBUR*), and the presented area, vulnerable area, and tracking error accumulation variables (*SUMAH, SUMAK, TRKSUM)* are set to zero.

Block 3: The distance between the target and round positions in rectangular coordinates at the approximate time of closest approach, t_{ap}, (variable *TAPRX*) is calculated via subroutine MATSUB and stored in variable *RPOS*.

Block 4: The round velocity relative to the target is calculated via matrix subtraction utility MATSUB and is stored in *RVEL*.

Block 5: The time period, t_0, subsequent to t_{ap} that is required to obtain the CPA is calculated using Equation [2.35-8] and is stored in *T0*.

Blocks 6 and 7: If the time to CPA is less than the minimum time of flight (TOF) set in the fire control computer (0.2 sec), the miss distance (MD) components and round and target positions at CPA are calculated. Miss distance components are calculated using the miss distance at the most recent simulation time step (*TAPRX*, also the approximate time of closest approach at this point), the time to CPA (*T0*), and the relative velocity of Block 4; Equation [2.35-2] is used to implement these calculations. The round and target positions at CPA are extrapolated from their current velocities and the time to CPA using Equation [2.35-1].

Block 8: The MD components between the round and target at CPA (*RPOS*) are copied to *XMIS, YMIS*, and *ZMIS*.

Block 9: The round and target positions at CPA are transformed from rectangular to polar coordinates via subroutine POLAR.

Block 10: Function AZDIFF calculates the difference between the round and target azimuths, and this azimuthal miss angle is stored in *AZMIS*. The difference in round and target elevation angles at CPA is calculated and stored in *ELMIS*. These two global variables are used to adjust the aim point by simulating tracer feedback during either optical tracking with speed rings or with a mechanical computing sight; the Probability of Hit FE does not utilize these variables.

Block 11: If the counter for the number of rounds fired (*SHLFIR*) is non-zero, rounds have been fired since the last tracer adjustment in Subroutine OPTMCS or SPDRNG. *SHLFIR* is set to *TOTFIR*, the total number of rounds fired for the engagement (a non-zero number).

Block 12: To assess the effects of dispersion only on the endgame calculations, the user may select a “perfect” FCC by inputting “P” as the FCC order in the input parameter file. In this case, the x, y, and z components of the miss distance (*XMIS, YMIS, ZMIS*) are set to zero.

Block 13: Function HYPOT calculates the miss distance as the square root of the sum of the squares of the x, y, and z miss distance components; this is described by Equation [2.35-9].

Block 14: The 85 and 100 mm guns have timed rounds which are set to explode at the estimated time of flight to the computed CPA. In *RADGUNS*, if a 85 or 100 mm gun is selected and the miss distance computed in Block 13 is less than 5 meters, the miss distance and its x, y, and z components are set to zero.

Block 15: Miss distance is stored in *OPTOUT(36)* for printout if selected as an option by the user.

Block 16: The relative position vector is copied to *RPOS2* using subroutine MATCPY. Subroutine MATSCL scales the relative velocity vector by -1.0 and stores it in *RVEL2*. *RVEL2* now stores the target velocity relative to the round.

Block 17: Subroutine ORIENT computes the target's angular orientation (variables *YAW*, *PITCH*, and *ROLL*) with respect to the weapon's coordinate frame. ORIENT uses the target orientation and its location (x, y, and z coordinates) to calculate the target azimuthal aspect angle presented to the weapon system (variable *PHI*) and the target elevation aspect angle presented to the weapon system (variable *THETA*).

Block 18: If the user has selected the Model View option, the relative position and velocity vectors along with the yaw, pitch, and roll of the aircraft in degrees are written to unit 21.

Block 19: Subroutine ROTATE transforms the relative position and velocity vectors from the weapon's coordinate system to the target's coordinate system. The target frame assumes straight and level flight from North to South, thus the vectors are rotated by an angle of *YAW-PI*, which will yield no rotation if the vector is already pointing North to South.

Block 20: The azimuth angle of *RVEL2* is computed (in the target frame of reference) as the arc tangent of the y component of the relative velocity vector over the x component. Function CIRCLE ensures that the azimuth angle is between 0 and 2π. The elevation angle of *RVEL2* is computed as the negative arc tangent of the z component of the relative velocity vector over the ground component (square root of the sum of the squares of the x and y components as computed by Function HYPOT).

Block 21: Subroutine ROTATE transforms the relative position vector from the target's coordinate system to the two dimensional plane of the projectile (perpendicular to the target center).

Block 22: *RPOS2(2)* now holds the horizontal component of the miss distance from the target center along the target path, while *RPOS2(3)* contains the vertical component. These components are stored in variables *DX* and *DY*.

Block 23: The approximate range to CPA is computed as the square root of the sum of the squares of the x, y, and z components of the round position vector (vector from weapon to round) at CPA, and is stored in *APRRNG*. The x and y components of the position vector are stored in *APRRGX* and *APRRGY*.

Block 24: If the user has specified the Model View option, the azimuth and elevation angles of the relative velocity vector in the target's coordinate frame and the range at CPA are written to unit21.

Block 25: The range of the round at CPA is saved for optional printout.

Block 26: If this is the first call to subroutine HITPRB, some variables need to be initialized. *FIRSTB*, *FIRSTD*, and *FIRSTZ* are used in "SNGL" (single fly-by) simulations to flag the first write to units 18 (burst-by-burst firing data file), 17 (round-by-round firing data file), and 19 (round-by-round firing data file for plotting burst probabilities) respectively. *FIRSTS* is used in "MULT" (multiple fly-by) simulations to flag the first shot. No rounds have been fired if this branch is executed, so the probability that all rounds fired have either missed or not killed the target is 1.0 (*PMH*, *PMK1*, *PMK2*, *PMK3*, *PMK4*). The slant range of the round at CPA for the first shot is set to the currently stored round range at CPA. *FIRST(8)*, the flag signaling the first call to HITPRB, is cleared.

Block 27: The standard deviation of the burst angular dispersion is calculated at the range of the round using Equation [2.35-41], and is stored in *SIGMA*. Function CONV converts angular dispersion in Soviet mils to radians.

The ellipsoidal width-to-height ratio for a given aspect angle can be interpolated from the ellipsoidal width-to-height ratios of the three relevant views (left/right, top/bottom, front/rear). The target's angular orientation was computed in Block 17. Blocks 28 - 44 implement the ellipsoidal ratio interpolation, where the relevant view area and ellipsoidal ratio data are stored in variables *TVH3* and *ERAT*, respectively.

If the azimuthal aspect angle (*PHI)* is greater than $\pi/2$ and less than 1.5π, the rear view of the aircraft is visible from the weapon system. The next three blocks implement rear-view data storage for use in future interpolation.

Block 28: For this range of azimuths, the ellipsoidal ratio for the rear view of the target is stored in *ERAT(1)* as the first ellipsoidal ratio used for interpolation purposes; this implements Equation [2.35-15].

Block 29: An azimuth interpolation factor, *AZFAC*, is calculated as the ratio of the absolute difference between the true target aspect and the straight rear view (*PHI-PI* for *PHI>PI*, *PI-PHI* for *PHI*$\leq$*PI*) to the maximum deviation from the rear view (*PI/2*). This implements Equation [2.35-14].

Block 30: The rear view presented/vulnerable area data is stored in *TVH3(1)* as the first of three relevant views. This implements Equation [2.35-16].

If *PHI* does not fall between $\pi/2$ and 1.5π the front view of the aircraft is visible from the weapon system. The next three blocks implement front-view data storage for use in future interpolation.

Block 31: In this case (of azimuth aspects), the ellipsoidal ratio for the front view of the target is stored in *ERAT(1)*. This implements Equation [2.35-18].

Block 32: The azimuth interpolation factor, *AZFAC*, is calculated using Equation [2.35-17].

Block 33: The front view presented/vulnerable area data is stored in *TVH3(1)* as the first of three relevant views. This implements Equation [2.35-19].

If *PHI* is less than π, the left side of the aircraft is visible. The next two blocks describe the determination of the second of three vulnerable/presented area and ellipsoidal ratio values to be used in the interpolation.

Block 34: In this case, the ellipsoidal ratio for the left side of the target is stored as the second ratio for use in the interpolation. This implements Equation [2.35-20].

Block 35: The left view presented/vulnerable area data is stored as the second of three relevant views. This implements Equation [2.35-21].

If *PHI* is equal to or greater than π, the right side of the aircraft is visible from the weapon system. The next two blocks describe the determination of the second of three vulnerable/presented area and ellipsoidal ratio values to be used in the interpolation.

Block 36: In this case, the ellipsoidal ratio for the right side of the target is stored as the second ratio for use in the interpolation. This implements Equation [2.35-22].

Block 37: The right view presented/vulnerable area data is stored as the second of three relevant views. This implements Equation [2.35-23].

If the elevation aspect angle *THETA* is equal to or less than $\pi/2$, part of the bottom of the target is exposed to the weapon system. The next three blocks describe the determination of the third of three vulnerable/presented area and ellipsoidal ratio values to be used in the interpolation.

Block 38: In this case, the ellipsoidal ratio for the bottom of the target is stored as the third ratio for use in the interpolation. This implements Equation [2.35-25].

Block 39: An elevation interpolation factor, *ELFAC*, is calculated as the ratio of the difference between the true target aspect and the bottom view (*THETA - 0*) to the maximum deviation from the bottom view (*PI/2*). This implements Equation [2.35-24].

Block 40: The bottom view presented/vulnerable area data is stored as the third of three relevant views. This implements Equation [2.35-26].

If *THETA* is equal to or greater than $\pi/2$, part of the top of the target is exposed to the weapon system. The next three blocks describe the determination of the third of three vulnerable/presented area and ellipsoidal ratio values to be used in the interpolation.

Block 41: In this case, the ellipsoidal ratio for the top of the target is stored as the third ratio for use in the interpolation. This implements Equation [2.35-28].

Block 42: An elevation interpolation factor, *ELFAC*, is calculated as the difference between the top view and the true aspect angle over the maximum deviation from the top view. This implements Equation [2.35-27].

Block 43: The top view presented/vulnerable area data is stored as the third of three relevant views. This implements Equation [2.35-29].

Block 44: Interpolation between the three width/height ratios is computed using Equation [2.35-30].

Block 45: The actual time of closest approach (*TCLOSE*) is calculated through addition of the estimated time of closest approach passed from subroutine SHOOT (*TAPRX*) and the estimated time to CPA calculated in Block 5 (*T0*).

Block 46: If no rounds are scheduled to be fired, execution resumes with the decision block following Block 69. Otherwise, the relative velocity vector, *RVEL*, is rotated to the weapon system coordinate frame via subroutine ROTATE. The weapon frame is *-(YAW-PI), PITCH, -ROLL* with respect to the target frame.

Block 47: The target areas (both hit and vulnerable) as viewed from the round are initialized to zero before interpolation begins in Block 50.

Block 48: If a "U1" or "U2" type target was specified by the user or a "1" was selected by the user as the number of target vulnerable/presented area views, single view data are stored in *AREAH, AREAK1, AREAK2, AREAK3,* and *AREAK4*.

Block 49: If the number of target views selected by the user is six, the total cross-sectional area of the target as seen by the round is calculated using Equation [2.35-31], and is stored in *AREAH*.

Block 50: If the number of vulnerable areas is not one or six, the user has specified twenty-six view data. Subroutine INTGTA is called to calculate the presented area (returned via argument *AREAH*) of the target by interpolating between three or four target views. Subroutine CONV converts the azimuth and elevation angles between target and round from radians to degrees. Ninety degrees is subtracted from the elevation angle to match the COVART coordinate convention required when calling INTGTA (see Figure 2.35-3). Subroutine INTGTA is described by Figure 2.35-9.

Block 51: The target areas presented to the round are saved for optional printout.

Block 52: Equation [2.35-13] is used to calculate the dimension of the major axis using the interpolated presented area and ellipsoidal ratio). Similarly, the minor axis dimension is found by solving Equation [2.35-11]. These are stored in *AH* and *BH*, respectively.

Block 53: *S2*, the variance of round dispersion at the target range is found by squaring *SIGMA* of Block 27.

Block 54: The standard deviation of the diffuse target presented area, S_X and S_y, are calculated using Equations [2.35-39] and [2.35-40], and are stored in *SXH* and *SYH*, respectively.

Block 55: The power of the exponential in Equation [2.35-45] is calculated and stored in *PTEMH*.

Block 56: The exponential of Equation [2.35-45] is evaluated based on the power computed in Block 55. If the power is less than -70.0, the exponential is set to zero (e^{-70} = 3.975 x 10^{-31}); otherwise, the exponential is evaluated. This is re-written to variable *PTEMH*.

Block 57: Finally, the single shot P_h defined by Equation [2.35-45] is evaluated using the exponential of Block 56; this FE output is stored in *PHIT*.

Block 58: The P_h value is used to generate optional print statistics. The number of expected hits for this scan is calculated as the product of the probability of one round hitting the target times the number of rounds fired this scan. The number of expected hits for the engagement is the summation of the number of expected hits for the current scan plus all previous scans; the cumulative value is stored in *EXPHTS*.

Block 59: The total number of expected hits is saved for optional printout.

Block 60: The probability that all rounds have missed the target is calculated using Equation [2.35-52], and is stored in *PMH*. If the probability of miss is less than $1x10^{-20}$, it is set to zero to avoid an underflow condition.

Block 61: The cumulative probability that *at least* one round has hit the target is calculated using Equation [2.35-53], and is stored in *PHCUM*.

Block 62: Similarly, the cumulative probability that all rounds in the current burst have missed the target is calculated as the product of the individual probabilities of miss for each round in the burst via Equation [2.35-52], and is stored in *PMHB*. As with cumulative, the burst probability of miss is checked to prevent an underflow condition.

Block 63: The probability that at least one round in the burst has hit the target is calculated using Equation [2.35-53].

Block 64: Variable *TOTFIR* holds the total number of rounds fired to this point in the engagement. *NUMSHL* holds the number of rounds fired this scan (or, equivalently, for this call to HITPRB). Thus, the number of the first round fired this scan (*NSHL1*) is one more than the total number of rounds fired to this point less the number of rounds fired this scan. The number of the last round fired this scan (*NSHL2*) is the number of rounds fired to this point.

Block 65: If no data are available for a given vulnerable area type, the single shot, burst, and cumulative P_h and probability of kill (P_k) kill are set to zero as well as the number of expected hits (also the optional presented area expected hits output, *OPTOUT(37)*).

Block 66: If the beginning or end of a burst or the last round in the magazine occurs, or the user has specified printout of the individual shots, subroutine MOVTAR is used to compute the current target position, velocity, and acceleration.

Block 67: Subroutine EVENT is called at the beginning of a burst to record a “Commence firing” event. The current time, target position, round number, single shot, burst, and cumulative probabilities of hit and kill are recorded for printout in the tabular report and are written to the terminal with the round miss distance and intercept range.

Block 68: If the user specified printout of individual shots in the input parameter file, subroutine EVENT is called to record a "Shot fired" event. The current time, target position, the number of the first round fired this scan, and the single shot, burst, and cumulative P_h/P_k values are recorded for printout in the tabular report. A "Shot fired" message is written to the terminal along with the current time, single-shot P_h, and cumulative P_h.

Block 69: If the current round is the last round in the magazine, subroutine EVENT is called to record a "Guns out of ammunition" event. The current time, target position, round number, single shot, burst, and cumulative probabilities of hit and kill are recorded for printout in the tabular report. A "Guns out of ammunition" message and the current simulation time are written to the terminal.

Block 70: If a "MULTC" or "MULTE" simulation type (multiple fly-by with different output options designated by "B", "C", or "E") has been selected by the user, subroutine WRTMUL is called with the time of closest approach and the current round number. For "MULTC" simulations, WRTMUL writes the target position at user specified cumulative P_h/P_k increments to units 26-65 (depending on the increment). For "MULTE" simulations, WRTMUL writes the target position at user specified expected hit increments. Units 22-25 contain target position at detection, successful initiation of track, and intercept by the first round fired.

Blocks 71 and 72: If the user has specified a "MULTB" simulation, subroutine WRTMUL writes the target position at the first shot and at the end of each burst to units 26-65. The first shot flag, *FIRSTS*, is cleared (set to false) after each write.

Block 73: If a multiple fly-by simulation was not chosen, the user specified a "SNGL" simulation. If the user has chosen to output firing data (either with the firing data or both scan and firing data option), miss distance is stored in *DMIS* so it is not overwritten.

Block 74: If the user selected the weapon's time frame for printout of time and range variables *(TIMEPR)*, the time to be printed in the tabular output file is set to the current time.

Block 75: If the first round fired this scan was also the first round fired during the engagement, the current time is saved as the time of the first shot (*TFS*).

Block 76: If the user did not select the weapon's time frame, the target's time frame will be used for printout of time and range. *TIMEPR* is set to the time of closest approach (*TCLOSE*).

Block 77: If the first round fired this scan is also the first shot in the engagement, the time at closest approach also is saved as the time of the first shot (*TFS*).

Block 78: The miss distance angle in milliradians is calculated using the definition of a dot product, and is stored in *AE*.

Block 79: Subroutine MOVTAR is called to calculate the target position, velocity, and acceleration at the print time set in either Block 74 or 76.

Block 80: The rectangular target position coordinates are transformed to polar coordinates via subroutine POLAR.

Block 81: When specifying output of shooting performance data files, the user also specifies the type of target range to be written to them (either slant range, ground range, or the x or y component of range). If the user specifies slant range, the range output by subroutine POLAR is saved as the range for printout.

Block 82: If the user specifies ground range to the target for printout, the square root of the sum of the squares of the x and y components of the target position (calculated via function HYPOT) is stored as the range for printout.

Block 83: If the user specifies the x component of target range, the x component of the target position is saved for printout.

Block 84: If the user specifies the y component of target range, the y component of the target position is saved for printout.

Block 85: Several quantities are averaged over a burst for use in the printout and calculation of burst P_h values via the Salvo Formula. *NSBUR*, the number of rounds fired in the current burst, is incremented by the number of rounds fired this scan (*NUMSHL*). The miss distance, miss distance angles, miss distances squared, shot times, target ranges, ranges to the round at CPA, presented and vulnerable areas of the target as seen by the round, and the tracking errors are accumulated for each round fired.

Blocks 86 - 95 are executed if the user has selected the “BURND” output option (unit 19, which records data on a round-by-round basis during a burst) and a new burst commences.

Block 86: If the user has selected the “BURND” output option, time and range (in the specified target/weapon time frame) are written on a round-by-round basis and probabilities of hit and kill for each burst are written. Zero probabilities of hit and kill are written just prior to and following each burst and negative signs are attached to data for egressing targets. This is done to plot $P_{h/k}$ as it accumulates during a burst. At the start of each burst, subroutine MOVTAR is called to calculate the target position 0.01 seconds prior to the start of the burst.

Block 87: If the user has specified output of target slant range, subroutine POLAR is called to convert the target's rectangular coordinates to azimuth, elevation, and slant range.

Blocks 88 - 90: User-specified target ranges (for a “BURND” output option) are calculated and stored as described in Blocks 82 - 84, except that the variable name is *RNG* rather than *RG*.

Block 91: If the user specified English units for printout, the range value is converted from meters to feet.

Block 92: If this is the first write to unit 19, a header is written to it.

Blocks 93 and 94: If this is not the first write to unit 19, zero burst P_h and P_k values are written 0.01 seconds before the difference between the print time and the time of the first shot. *FIRSTZ*, the flag signaling the first write to unit 19, is cleared.

Block 95: If the first round in a burst has already been fired for a "BURND" simulation, subroutine MOVTAR is called to calculate target position at the print time of each shot.

Block 96: Subroutine POLAR converts the target position in rectangular coordinates to polar coordinates.

Blocks 97 - 100: The user-specified target range (after the call to MOVTAR in Block 95) is stored in variable *RNG* as described in Blocks 81 - 84.

Block 101: Function RADVEL computes the radial velocity of the target with respect to the weapon system. The range from the radar will be positive regardless of target orientation from the radar. In order to distinguish between targets flying toward the weapon system and targets flying away, a negative sign is attached to the burst probability data for targets with positive radial velocities (flying away).

Block 102: If the user selects only the x or y component of range, a negative sign attached to the burst probability data will alert the user to the target's orientation with respect to the radar instead of a negative sign attached to the burst probability data.

Block 103: If the user selected the output in English units, the target range and round miss distance in meters are converted to feet.

Blocks 104-108 are executed if the user has selected output to unit 17 (either through the "ROUND" or "ALL" file output option) and rounds have not been fired during the current scan. Firing data are written on a round-by-round basis.

Block 104: If unit 17 has not previously been written to, a header is written to the output file.

Block 105: Variable *TOTFIR* is passed from subroutine SHOOT and represents the total number of rounds fired from the beginning of the engagement to the end of the current scan. The number of rounds fired this scan is subtracted from *TOTFIR* such that *TOTFIR* now represents the number of the last round fired during the last scan.

Blocks 106 and 107 are executed once for each round fired this scan.

Block 106: *TOTFIR* is incremented so that it contains the current round number.

Block 107: The time of the current shot since the start of firing, the user specified target range, the number of the current round, the miss distance, the single shot and cumulative P_h values, the number of expected hits, and the single shot and cumulative P_k values for the four kill types are written to unit 17.

Block 108: Variable *FIRSTD*, the first write to unit 17 flag, is cleared.

Block 109: If the user specified the "BURND" firing output option, the time of each shot since the first round fired, the target range, and the signed burst probabilities (both hit and kill) are written to unit 19 for each round.

Blocks 105 - 119 are executed at the end of every burst for the "BURST" or "ALL" output options only.

Block 110: The Salvo Equation calculates burst $P_{h/k}$ based on the average (for the burst) target presented/vulnerable area, standard deviation of miss distance, the average ballistic dispersion, and the average miss distance for the number of rounds in the burst. Average values for target range, range to CPA, time, miss distance, and miss distance angle are found for the burst by dividing the summations of Block 85 by the number of rounds in the burst.

Block 111: The average ballistic dispersion radius is calculated by applying the angular dispersion of the rounds to the average range to CPA. Function CONV converts the angular dispersion in Soviet mils to radians.

Block 112: The standard deviation of miss distance for a burst is calculated using Equation [2.35-55].

Block 113: The average azimuth, elevation, and range tracking errors and the average target presented and vulnerable areas for the burst are calculated by dividing the cumulative values from Block 85 by the number of rounds in the burst.

Block 114: The average target presented/vulnerable areas, the standard deviation of the burst miss distances, the average ballistic dispersion, the average miss distance, and the number of rounds in the burst are passed to subroutine SALVO for the calculation of burst $P_{h/k}$ using Equation [2.35-46]; see the next flow diagram for a detailed description of SALVO.

Block 115: If the user selected the output in English units, the average target range, range shooting error, miss distance, and standard deviation of miss distance are converted from meters to feet.

Block 116: If this is the first burst in the engagement, a header is written to unit 18.

Block 117: The difference in time since the first shot and the time average of the burst, average target range for the burst, average azimuth, elevation, and range shooting errors, average miss distance, average miss distance angle, standard deviation of miss distance, burst $P_{h/k}$, and round probabilities of hit and kill are written to unit 18 at the end of each burst.

Block 118: If any of the Salvo probabilities are less than zero or greater than one, a warning is written to unit 18.

Block 119: *FIRSTB,* the flag signaling the first burst in an engagement, is cleared.

Blocks 120 - 126 are executed at the end of each burst for the "BURND" and "ALL" output options.

Block 120: If the user has specified the "BURND" output option, zeros are written to unit 19 following each burst. Subroutine MOVTAR calculates the target position 0.01 seconds after the last round in the burst is fired.

Blocks 121 - 124: Target range is determined for the new target position by exactly the same algorithm as described in Blocks 87 - 90.

Block 125: If the user has requested the output in English units, the target range is converted to feet.

Block 126: At 0.01 seconds following the last round in the burst, less the time of the first shot, the target range and zero probability values are written to unit 19.

Block 127: Execution returns to subroutine SHOOT.

Subroutine SALVO

Subroutine SALVO calculates the burst P_h using Equation [2.35-46] which is repeated below.

$$P_{h_{BUR}} = \sum_{i=1}^{N} \binom{N}{i} (-1)^{i+1} \left(\frac{A}{2\pi\sigma_b^2 + A} \right)^{i-1} \left(\frac{A}{2\pi(i\sigma_a^2 + \sigma_b^2) + A} \right) \exp\left(\frac{-i\pi D^2}{2\pi(i\sigma_a^2 + \sigma_b^2) + A} \right)$$

The subroutine flow diagram of Figure 2.35-7 decomposes Subroutine SALVO into numbered blocks, each containing a specific function in the routine. Variable names are enclosed in parentheses. A discussion of the implementation of each block follows the diagram.

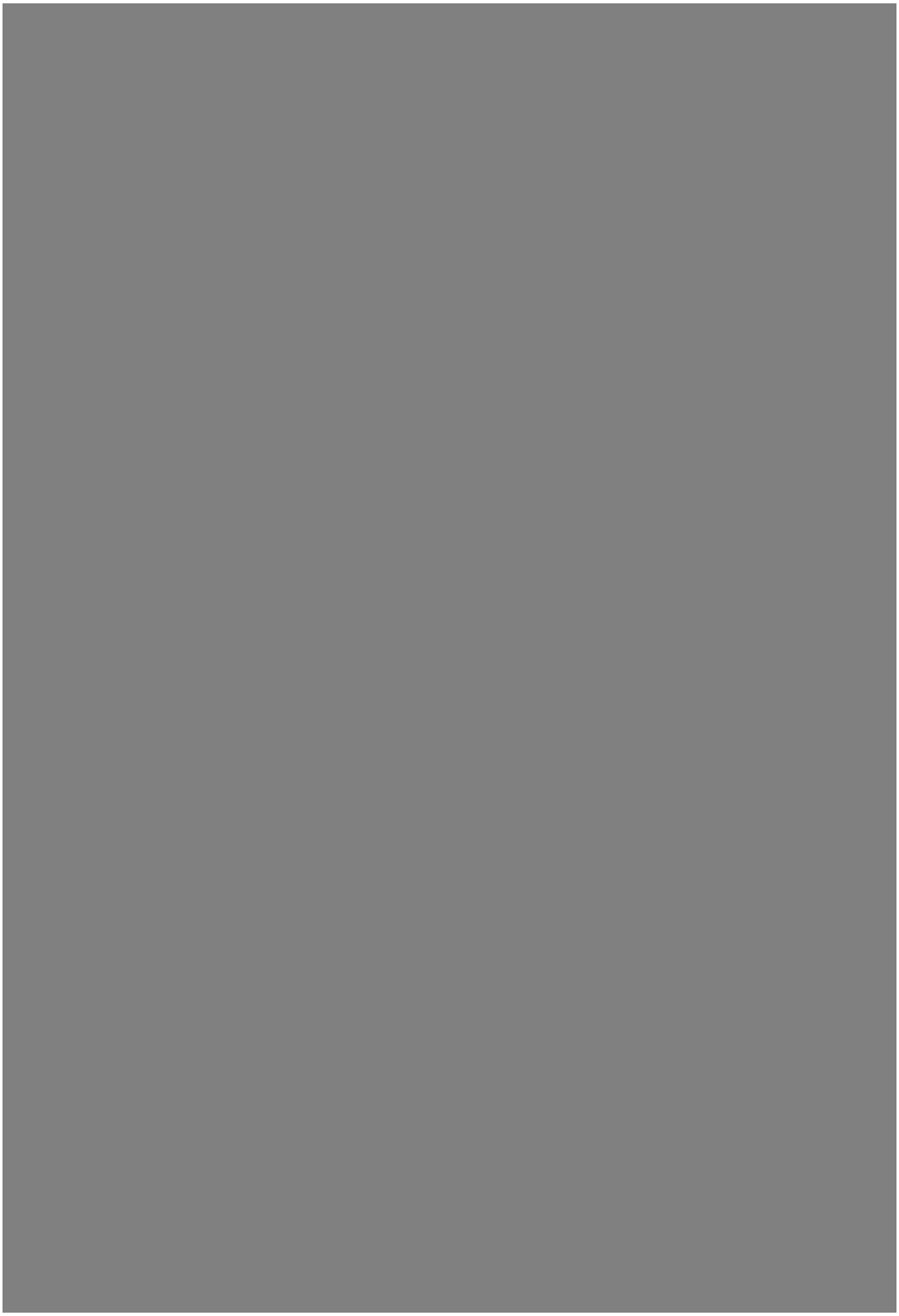

FIGURE 2.35-7. Subroutine SALVO Flow Chart.

The average target presented/vulnerable areas for the burst (*AVGH, AVGK1-AVGK4*), the standard deviation of the burst miss distances (*SDEVMD*), the ballistic dispersion applied at the average intercept range (*DISPD*), the average miss distance (*AVGMD*), and the number of rounds in the burst (*NSBUR*) are passed to Subroutine SALVO and stored in local variables *AV, ASIGMA, BSIGMA, D, NB,* and *PB,* respectively.

Block 1: Several variable declarations and DATA statements are executed using double precision to maintain accuracy in the calculations for large bursts. Real variables *AV, ASIGMA, BSIGMA,* and *D* are copied to double precision variables *DAV, DASIGM, DBSIGM,* and *DD.* Variables *DPB, DPBL,* and *PB* are initialized to zero. Variable *X* is used to store the combination calculations of the Salvo Equation; it is initialized to one.

If the average presented/vulnerable area of the target is less than 10^{-6}, program execution returns to subroutine HITPRB with a P_h of zero. If the presented/vulnerable area of the target is at least 10^{-6}, execution takes one of two paths based on burst size. If the burst consists of more than 50 rounds, the P_h is approximated based on the first 50 rounds in the burst; otherwise, the actual number of rounds in the burst is used to calculate the P_h for the burst.

The burst probability is the product of several intermediate calculations. As burst size increases, some calculations yield very large or very small numbers, which when multiplied together give inaccurate results. A 50-round burst was empirically derived as the limit which prevents the intermediate calculations from becoming excessively large or small; it applies over a wide range of target presented areas, miss distances, and dispersion radii.

Block 2: If the burst size is greater than 50 rounds, the ratio of the actual number of rounds to 50 rounds is calculated and stored in *DFACT*; this value will be used in Block 15 to scale the P_h for 50 rounds to estimate P_h for a larger burst.

Block 3: The burst size (JNB) is set to 50 rounds.

Blocks 4 and 5: If the burst size is 50 or less rounds, *DFACT* is set to one and the burst size is set to the actual number of rounds in the burst.

Blocks 6 through 12 are repeated once for each round until either the end of the burst occurs or the difference in successive calculations is less than 1.0×35^{-6} (double precision).

Block 6: Each combination (the first factor) in Equation [2.35-46] is calculated based on the preceding combination in the summation as described by Equation [2.35-49], which is stored in variable *X*.

Blocks 7 and 8:The power of the exponential of Equation [2.35-46] is calculated and stored in variable *DEXPON*. If the power is less than -170, it is set to -170 to prevent an underflow condition.

Block 9: The first three factors of Equation [2.35-46] are calculated in two steps; their product is stored in variable *TEMP1*.

Block 10: The term preceding the exponential expression in Equation [2.35-46] is calculated and stored in *TEMP2*.

Block 11: The exponential expression (final term) of Equation [2.35-46] is evaluated and stored in *TEMP3*.

Block 12: The results of the last three blocks are multiplied and then added to the summation variable *DPB*; the final value of the summation implements the Salvo Equation using Equation [2.35-46] for 50 rounds or less. Block 14 scales DPB if more than 50 rounds were fired.

Block 13: If the cumulative P_h for the burst has changed by at least 10^{-9} (double precision), the new probability is saved in *DPBL* as the former probability in the next comparison. Blocks 6 through 13 are repeated until the number of terms calculated equals the number of rounds in the burst. If the calculated value has not changed by at least 10^{-9}, the summation loop is terminated and execution continues with Block 14.

Block 14: The probability that at least one round in the actual number of rounds fired has hit the target is found by first applying the exponent of Blocks 2 or 4 (*DFACT*) to the probability of miss; probability of miss is calculated using the burst P_h from Block 12 using Equation [2.35-52]; this quantity is then subtracted from 1 to find the burst P_h, which implements Equation [2.35-53]. Note that this block does not change the result of variable *DPB* from Block 12 if 50 or less rounds were fired (because the exponent *DFACT* equals 1 in this case).

Block 15: The burst probability (*DPB*) is stored in real variable *PB* for return to HITPRB.

Subroutine INTGTA

Subroutine HITPRB calls Subroutine INTGTA to calculate the target presented area for a 26-view data format. Figure 2.35-8 depicts the logic flow and is followed by block descriptions.

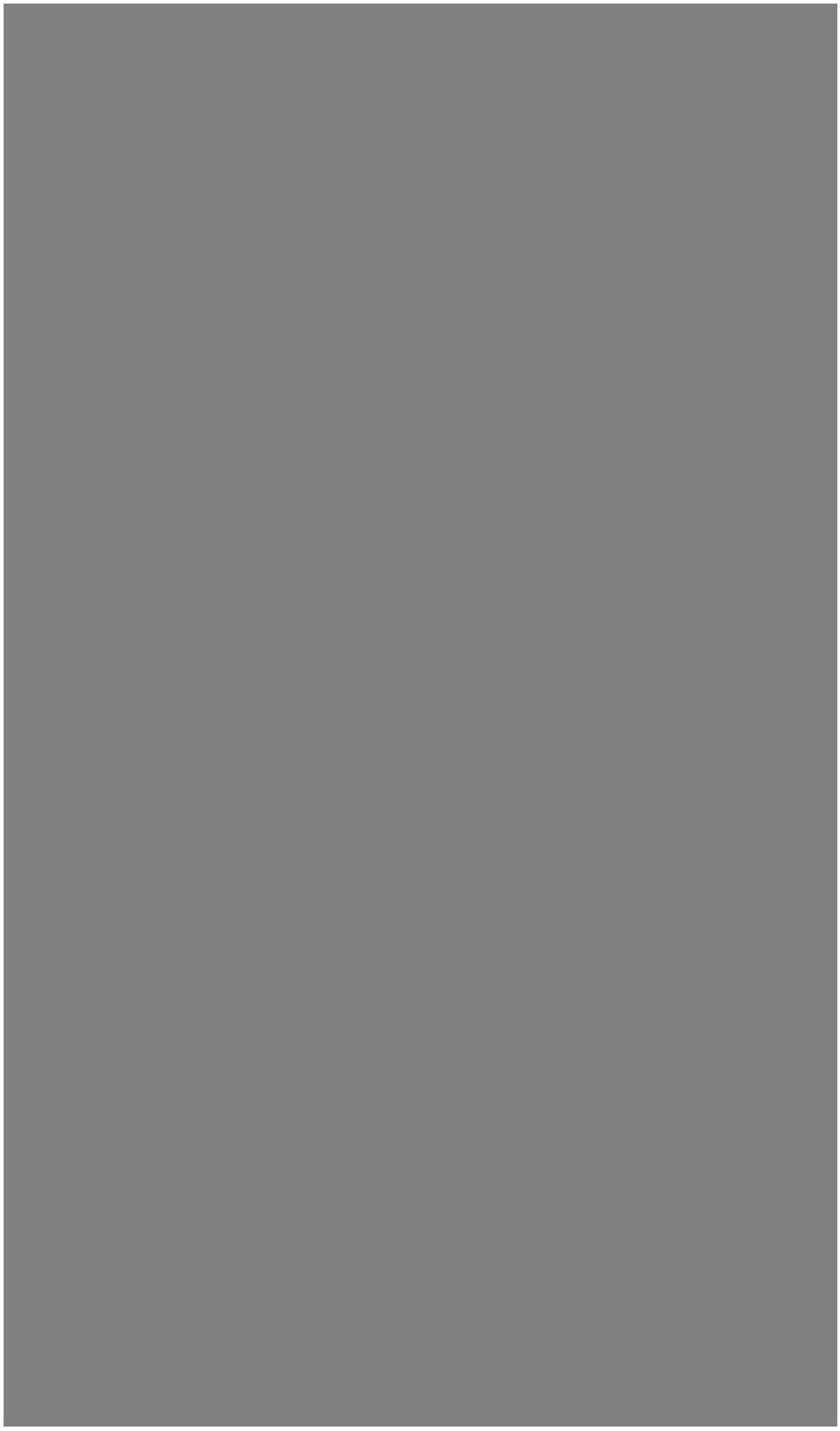

FIGURE 2.35-8. Subroutine INTGTA Flow Diagram.

Subroutine INTGTA interpolates between the 26-view target presented area data to determine the target area presented to the round. Interpolation is based on the azimuth and elevation aspect angles of the target. COMMON variables *TVH26(26)*, *TVK126(26)*, *TVK226(26)*, *TVK326(26)*, and *TVK426(26)* contain the target presented and vulnerable areas for 26 views of the target in the convention shown in Figure 2.35-3 and listed in Table 2.35-1.

The presented/vulnerable area data file has area data in 45 degree increments from latitudes (elevations) of -90 to 90 degrees, and longitudes (azimuths) of 0 to 315 degrees. The target elevation and azimuth aspects viewed from the round have corresponding area data at increments above and below them (unless the view is on a data increment). Linear interpolation in elevation yields an area at each of the two azimuth values. The presented area is then calculated by linear interpolation between the two azimuth areas values.

The interpolation scheme is implemented in part by calculation of indices that are array pointers to presented/vulnerable area data at an orientation close to the target orientation with respect to the round. Twenty-six cross-sectional views are characterized by eight azimuth and five elevation values. Tables 2.35-3 and 2.35-4 list the indices associated with the azimuth and elevation aspects, respectively.

TABLE 2.35-3. Azimuth View Indices.

Azimuth	View Index
180	0
225	1
270	2
315	3
0	4
45	5
90	6
135	7

TABLE 2.35-4. Elevation View Indices.

Elevation	View Index
-90	1
-45	2
0	35
45	18
90	26

Block 1: The lower azimuth and elevation indices (*NAZ1* and *NEV1*) are initially set to 0 and 35; this corresponds to the rear view of the target since an azimuth index of 0 is for the 180 degree azimuth aspect (rear), and an elevation index of 35 is for the zero degree aspect (level).

Block 2: If the target azimuth aspect angle is negative, it is converted to its equivalent positive angle. The azimuth interpolation factor is calculated using Equation [2.35-32], and is stored in *AZIFAC*. The elevation interpolation factor is calculated using Equation [2.35-33], and is stored in *ELVFAC*.

Blocks 3 through 8 involve determination of the elevation indices at two azimuth values; the lower and upper indices are stored in *NEV1* and *NEV2*, respectively.

Block 3: The lower elevation index was initialized to 35, corresponding to an elevation angle of 0 degrees. If the elevation angle is less than 0, the upper index is set to 2. These indices bracket an angle between 0 and -45 degrees.

Block 4: If the elevation angle is less than or equal to -45 degrees, the upper index is set to 1 and the lower index is set to 2 corresponding to an angle between -45 and -90 degrees.

Block 5: If the elevation angle is less than or equal to -90 degrees, the lower index is set to 1 (the upper index remains at 1).

Block 6: If the elevation angle is greater than or equal to 0 degrees, the upper index is set to 18. The lower index was initialized to 35 so the indices bracket an angle between 0 and 45 degrees.

Block 7: If the elevation angle is greater than or equal to 45 degrees, the upper index is set to 26 and the lower index is set to 18 corresponding to an angle between 45 and 90 degrees.

Block 8: If the elevation angle is greater than 90 degrees, the lower index is set to 26 (the upper index remains at 26).

Blocks 9 and 35 involve determination of the two azimuth indices that bound the given (input) azimuth aspect; these two azimuth indices are stored in *NAZ1* and *NAZ2*.

Block 9: The indexing convention has the lowest index (equal to zero) at a 180 degree azimuth aspect, and each index increments by one with a clockwise change of 45 degrees; thus, 180 degrees is first added to the given azimuth, and this is divided by the increment size (45 degrees) to obtain the first index. The indices range from 4 to 11 at this point, so to obtain values in the range from 0 to 7, a check is performed to determine if the index is greater than 7; if so, 8 is subtracted from the index.

Block 10: The second index is found by the same method as in Block 9, but with 45 degrees added to the given azimuth angle; this effectively adding one to the index before the limit check is performed.

Blocks 11 and 12: If the elevation angle is less than or equal to -45 degrees (*NEV2 = 1*), the bottom view of the target will be used as one of the four interpolation points. Similarly, if the elevation angle is equal to or exceeds 45 degrees (*NEV2 = 26*), the top view of the target will be used as one of the four interpolation points. If the elevation angle is equal to -90 or 90 degrees (*NEV1 = 1* or *NEV1 = 26*), area will be interpolated from three points. If the elevation angle is between -45 and 45 degrees, the array pointers (*NDX1* and *NDX2*) to the areas corresponding to the two elevations above and below the given elevation aspect

at the lower azimuth value are found by summing the lower azimuth index with each of the elevation indices.

Blocks 13 and 14: Using the array pointers calculated above, the target presented areas at the elevations surrounding the given elevation aspect angle at the lower azimuth value are stored in *TPA1* and *TPA2*. For an elevation angle of greater than or equal to 90 degrees, or less than or equal to -90 degrees, these areas will be equal.

Block 15: The elevation factor (*ELVFAC*) calculated in Block 2 is used to interpolate between presented areas at elevation values at the lower azimuth value; this interpolation is found using Equation [2.35-34], and is stored in *TPAZ1*.

Blocks 16 - 20: The same process described in Blocks 11 - 15 is used to interpolate between elevation values at the upper azimuth value. This interpolation implements Equation [2.35-35], and is stored in *TPAZ2*.

Block 21: The azimuth factor (*AZIFAC*) calculated in Block 2 is used to interpolate between the presented areas found in Blocks 15 and 20 using Equation [2.35-36]. The result is stored in *AREAH*, which is the target area presented to the round.

Blocks 22 - 28: Blocks 13 - 15 and 18 - 20 are repeated for each of the kill types. These are not included in the Probability of Hit FE.

The target presented and vulnerable areas are returned to subroutine HITPRB via arguments *AREAH, AREAK1, AREAK2, AREAK3,* and *AREAK4* for use in the P_h (and P_k) calculations.

Probability of Hit Inputs and Outputs

Table 2.35-5 identifies the output of the Probability of Hit FE. User inputs that effect the FE are described in Table 2.35-6. Local variables are not part of the inputs to the FE.

TABLE 2.35-5. Functional Element Output.

Variable Name	Description
PHBUR	Cumulative probability of hit
PHIT	Single round probability of hit
PHSAL	Burst probability of hit, Salvo Equation

TABLE 2.35-6. User Inputs Included in FE.

Variable Name	Variable Options	Description
FCCORN	1, 2, P	First Order, Second Order, Perfect Fire Control, respectively. "P" sets miss distance to zero.
NOAREA	1, 6, 26	Number of target presented/vulnerable areas to be used.
VASCL	Any Number	Scale factor to the existing presented/vulnerable area data.

Subroutine HITPRB is the module which implements the P_h calculations. The inputs and output of HITPRB are identified in Table 2.35-7. Subroutines INTGTA and SALVO are called by HITPRB to calculate the 26-view target presented area and P_h for a burst; inputs and outputs for these two modules are provided in Tables 2.35-12 and 2.35-17. The remainder of the tables list the inputs and outputs of mathematical utility routines that support the Probability of Hit FE. Variables listed in bold letters denote those which directly implement the FE.

TABLE 2.35-7. Subroutine HITPRB Inputs and Outputs.

SUBROUTINE: HITPRB					
Inputs			**Outputs**		
Name	**Type**	**Description**	**Name**	**Type**	**Description**
CALIBR	Common	Weapon system caliber (mm)	**APRFS**	Common	Range of first round at CPA (m)
DISPER	Common	Angular ballistic dispersion (mils)	**APRRGX**	Common	x component of range of round at CPA (m)
EBURST	Common	End of burst flag	**APRRGY**	Common	y component of range of round at CPA (m)
FCCORN	Common	Order of FCC	**AZMIS**	Common	miss azimuthal angle magnitude (rad)
FIRST	Common	First call flag	**ELMIS**	Common	miss elevation angle magnitude (rad)
LSTSHL	Common	Last round flag	**ERATIO**	Common	Width/height ratio of ellipsoidal target
MVUON	Common	Model view graphics option	EXPHTS	Common	Number of expected hits
NOAREA	Common	Number of target vulnerable areas	OPTOUT	Common	Storage for printed variables
NUMSHL	Argument	Rounds fired this scan	**PHBUR**	Common	Cumulative P_h for burst
PI	Common	3.14159265359 (rad)	PK1BUR	Common	Cumulative P_k for burst - Type 1
PRTCOD	Common	Output option - scan or firing data or both	PK2BUR	Common	Cumulative P_k for burst - Type 2
PRTTFR	Common	Time frame for firing option	PK3BUR	Common	Cumulative P_k for burst - Type 3
PRTRNG	Common	Range type for firing option	PK4BUR	Common	Cumulative P_k for burst - Type 4

TABLE 2.35-7. Subroutine HITPRB Inputs and Outputs. (Contd.)

SUBROUTINE: HITPRB					
Inputs			**Outputs**		
Name	**Type**	**Description**	**Name**	**Type**	**Description**
PRTTYP	Common	Type of firing data files written for "SNGL" simulation	**PHCUM**	Common	Cumulative P_h for engagement
PRTSHT	Common	Print individual shots flag	PK1CUM	Common	Cumulative P_k for engagement - Type 1
PRTUNT	Common	File designators for shooting data option	PK2CUM	Common	Cumulative P_k for engagement - Type 2
SBURST	Common	First round in burst flag	PK3CUM	Common	Cumulative P_k for engagement - Type 3
SHELL	Argument	Round position, velocity, and acceleration (m, m/s, m/s^2)	PK4CUM	Common	Cumulative P_k for engagement - Type 4
SIMTYP	Common	Simulation type - SNGL, MULT, DETR, RCSP	**PHIT**	Common	Single shot P_h
T	Argument	Simulation time (sec)	PKIL1	Common	Single shot P_k - Type 1
TAPRX	Argument	Approximate time of closest approach (sec)	PKIL2	Common	Single shot P_k - Type 2
TARGET	Argument	Target position, velocity, and acceleration (m, m/s, m/s^2)	PKIL3	Common	Single shot P_k - Type 3
TOTFIR	Argument	Total rounds fired	PKIL4	Common	Single shot P_k - Type 4
TVH6	Common	Target presented areas - 6 views	**SHLFIR**	Common	Rounds fired since last call to SPDRNG
TVK16	Common	Target vulnerable areas - 6 views, Type 1 kill	**TRKERR**	Common	AZ, EL, RG tracking errors mrad m)
TVK26	Common	Target vulnerable areas - 6 views, Type 2 kill	**XMIS**	Common	x component of miss distance (m)
TVK36	Common	Target vulnerable areas - 6 views, Type 3 kill	**YMIS**	Common	y component of miss distance (m)
TVK46	Common	Target vulnerable areas - 6 views, Type 4 kill	**ZMIS**	Common	z component of miss distance (m)
TWOPI	Common	2π			
VADATA	Common	Target presented/vulnerable area data available switch, character variable			

TABLE 2.35-8. Subroutine AZDIFF Inputs and Outputs.

Subroutine: AZDIFF					
Inputs			**Outputs**		
Name	**Type**	**Description**	**Name**	**Type**	**Description**
AZ1	Argument	First azimuth angle (rad)	**AZDIFF**	Common	Difference between two azimuth angles (rad)
AZ2	Argument	Second azimuth angle (rad)			
PI	Common	π			
TWOPI	Common	2π			

TABLE 2.35-9. Subroutine CONV Inputs and Outputs.

Subroutine: CONV					
Inputs			**Outputs**		
Name	**Type**	**Description**	**Name**	**Type**	**Description**
PI	Common	π	**CONV**	Function	Value of angle in new units
X	Argument	Angle to be converted			
UNITS	Argument	Character variable specifying type of conversion			

TABLE 2.35-10. Subroutine DOT Inputs and Outputs.

Subroutine: DOT					
Inputs			**Outputs**		
Name	**Type**	**Description**	**Name**	**Type**	**Description**
N	Argument	Dimension of VEC1 and VEC2	**DOT**	Function	DOT product of VEC1 and VEC2
VEC1	Argument	First input vector			
VEC2	Argument	Second input vector			

TABLE 2.35-11. Subroutine HYPOT Inputs and Outputs.

Subroutine: HYPOT					
Inputs			**Outputs**		
Name	**Type**	**Description**	**Name**	**Type**	**Description**
X	Argument	X Coordinate (m)	**HYPOT**	Function	Distance from origin to the point defined by (X, Y, Z) (m)
Y	Argument	Y Coordinate (m)			
Z	Argument	Z Coordinate (m)			

TABLE 2.35-12. Subroutine INTGTA Inputs and Outputs.

Subroutine: INTGTA					
Inputs			**Outputs**		
Name	**Type**	**Description**	**Name**	**Type**	**Description**
AZI	Argument	Azimuth angle from target to round in target coordinate system (deg)	**AREAH**	Argument	Target area presented to round (m^2)
ELV	Argument	Elevation angle from target to round in target coordinate system (deg)	AREAK1	Argument	Target vulnerable area, Kill Type 1 (m^2)
TVH26	Common	26-view presented area data (m^2)	AREAK2	Argument	Target vulnerable area, Kill Type 2 (m^2)
TVK126	Common	26-view vulnerable area data for Type 1 P_k (m^2)	AREAK3	Argument	Target vulnerable area, Kill Type 3 (m^2)
TVK226	Common	26-view vulnerable area data for Type 2 P_k (m^2)	AREAK4	Argument	Target vulnerable area, Kill Type 4 (m^2)
TVK326	Common	26-view vulnerable area data for Type 3 P_k (m^2)			
TVK426	Argument	26-view vulnerable area data for Type 4 P_k (m^2)			

TABLE 2.35-13. Subroutine MATSUB Inputs and Outputs.

Subroutine: MATSUB					
Inputs			**Outputs**		
Name	**Type**	**Description**	**Name**	**Type**	**Description**
MAT1	Argument	Input matrix	**RES**	Argument	Matrix, result of subtracting MAT2 from MAT1
MAT2	Argument	Input matrix			
IMAX	Argument	Dimension of matrix rows			
JMAX	Argument	Dimension of matrix columns			

TABLE 2.35-14. Subroutine ORIENT Inputs and Outputs.

Subroutine: ORIENT					
Inputs			**Outputs**		
Name	**Type**	**Description**	**Name**	**Type**	**Description**
TARGET	Argument	Target's position (m), velocity (m/s) and acceleration (m/s^2)	**OPTOUT**	Common	Storage for printed variables
BANGLE	Common	Bank angle (rad) (equal to roll)	**PHI**	Argument	Target azimuthal aspect angle presented to weapon system (rad)
BLUMAX	Common	Block of target flight path data in Bluemax format	**PITCH**	Argument	Target pitch angle (rad)
MOVTYI	Common	Type of target flight path, renamed based on user input	**ROLL**	Argument	Target roll angle (rad)
MOVTYP	Common	User-input, type of target flight path	**THETA**	Argument	Target elevation aspect angle presented to weapon system (rad)
PI	Common	π			

TABLE 2.35-15. Subroutine POLAR Inputs and Outputs.

Subroutine: POLAR					
Inputs			**Outputs**		
Name	**Type**	**Description**	**Name**	**Type**	**Description**
RECT	Argument	Rectangular coordinates of a point (m)	**AZ**	Argument	Polar coordinate, azimuth (rad)
			EL	Argument	Polar coordinate, elevation (rad)
			RANGE	Argument	Range from origin to point of interest (m)

TABLE 2.35-16. Subroutine ROTATE Inputs and Outputs.

Subroutine: ROTATE					
Inputs			**Outputs**		
Name	**Type**	**Description**	**Name**	**Type**	**Description**
VEC	Argument	Vector to be transformed to another coordinate frame	**VEC**	Argument	Transformed vector
PITCH	Argument	Pitch angle in weapon frame, defines Y axis in new frame (rad)			
ROLL	Argument	Roll angle in weapon frame, defines Z axis in new frame (rad)			
YAW	Argument	Yaw angle in weapon frame, defines X axis in new frame (rad)			

TABLE 2.35-17. Subroutine SALVO Inputs and Outputs.

Subroutine: SALVO					
Inputs			**Outputs**		
Name	**Type**	**Description**	**Name**	**Type**	**Description**
AV	Argument	Average target vulnerable/presented area for burst (m^2)	**PB**	Argument	Burst probability of hit/kill using Salvo Equation
ASIGMA	Argument	Standard deviation of burst miss distances (m)			
BSIGMA	Argument	Ballistic dispersion applied at range of closest point of approach (m)			
D	Argument	Average miss distance for burst			
NB	Argument	Number of rounds in burst			

2.35.4 Assumptions and Limitations

This section presents the assumptions and limitations associated with the modeling of P_h in RADGUNS for the applicable system.

The target area distribution is assumed to be elliptic Gaussian with variances (S_x^2, S_y^2). The presented area of the elliptical target is determined relative to its centroid.

Miss distance is defined as the distance between the projectile (in its trajectory) and the target centroid at their closest point of approach.

When the round and target are closest to each other, the target center is in a plane perpendicular to the round's velocity vector (relative to the target). A Cartesian coordinate system can be imposed on this plane with the origin at the target centroid, and the round trajectory intersecting the plane at (x, y). The actual location of the round on the plane is distributed as bivariate independent Gaussian with means (X, Y), and variances (σ_x^2, σ_y^2). The probabilities associated with the target and round are assumed to be independent.

An assumption of the Salvo Formula is that a bivariate independent Gaussian distribution of possible aim points is caused by random tracking and aiming errors rather than a discrete aim point error for each projectile fired. Also, the P_h for each round in a multiple round engagement is assumed to be independent of probabilities of hit of the other rounds.

The target's location and orientation with respect to the weapon site is used to define the aspect angles THETA and PHI presented to the weapon system; these angles are then used to determine the presented target area. Thus, the assumption is that the round travels in a line between the weapon system and the target. The target presented area is then the area on a plane orthogonal to the weapon line of sight at the closest point of approach rather than on a plane orthogonal to the flight path of the round.